Imperial College of Science, Technology and Medicine
Department of Electrical and Electronic Engineering

Novel Applications and Contexts for the Cognitive Packet Network

Olumide Akinwande

Supervised by Professor Erol Gelenbe

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Electrical and Electronic Engineering
of Imperial College, November 2018

Abstract

Autonomic communication, which is the development of self-configuring, self-adapting, self-optimising and self-healing communication systems, has gained much attention in the network research community. This can be explained by the increasing demand for more sophisticated networking technologies with physical realities that possess computation capabilities and can operate successfully with minimum human intervention. Such systems are driving innovative applications and services that improve the quality of life of citizens both socially and economically. Furthermore, autonomic communication, because of its decentralised approach to communication, is also being explored by the research community as an alternative to centralised control infrastructures for efficient management of large networks.

This thesis studies one of the successful contributions in the autonomic communication research, the Cognitive Packet Network (CPN). CPN is a highly scalable adaptive routing protocol that allows for decentralised control in communication. Consequently, CPN has achieved significant successes, and because of the direction of research, we expect it to continue to find relevance. To investigate this hypothesis, we research new applications and contexts for CPN.

This thesis first studies Information-Centric Networking (ICN), a future Internet architecture proposal. ICN adopts a data-centric approach such that contents are directly addressable at the network level and in-network caching is easily supported. An optimal caching strategy for an information-centric network is first analysed, and approximate solutions are developed and evaluated. Furthermore, a CPN inspired forwarding strategy for directing requests in such a way that exploits the in-network caching capability of ICN is proposed. The proposed strategy is evaluated via discrete event simulations and shown to be more effective in its search for local cache hits compared to the conventional methods.

Finally, CPN is proposed to implement the routing system of an Emergency Cyber-Physical System for guiding evacuees in confined spaces in emergency situations. By exploiting CPN's QoS capabilities, different paths are assigned to evacuees based on their ongoing health conditions using well-defined path metrics. The proposed system is evaluated via discrete-event simulations and shown to improve survival chances compared to a static system that treats evacuees in the same way.

Acknowledgements

First of all, I would like to thank my supervisor, Professor Erol Gelenbe, for his priceless guidance and support. It has been a real privilege and honour to work and learn under his expert supervision. I am grateful for his patience, motivation, and knowledge which have benefitted me in all the time of my research.

I would like to thank my examiners, Dr. Aldo Faisal, and Dr. Avgoustinos Filippoupolitis for their helpful comments and encouragement.

I also appreciate the inspirational friends I made during my research. I would like to thank Sam, Akinbiyi, Tayo, Eniola, Tofade, Mihajlo, Huibo, Yonghua, Yuanchen, Lan, Taylor, Yasin, Elif, Adunola, John, Omer and Daniel for their sincere friendship.

Most of all, I would like to thank my parents, Engr. and Mrs. Akinwande, and my sister, Dr. Omolola Odigwe, for coming along with me on this journey. Their words of encouragement helped me to endure the most difficult parts of this process. My sincere thanks also go to my families in London, Wigan, and Bristol for their support and for enriching my experience in the United Kingdom.

Dedication

Dedicated to my beloved parents and sister, Engr. and Mrs. Akinwande and Dr. Omolola Odigwe.

Declaration of Originality

I, Olumide Akinwande, declare that the work presented in this thesis titled, "Novel Applications and Contexts for the Cognitive Packet Network", is my own. All the material in the thesis which is not my own work has been properly referenced and acknowledged.

Copyright declaration

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.

Contents

List of Tables

List of Figures

List of Abbreviations

ACKs acknowledgement packets

CDNs Content Delivery Networks

CPN Cognitive Packet Network

CPS Cyber-Physical Systems

CR content router

CS Content Store

DPs dumb packets

DV Distance Vector

FIB Forwarding Information Base

ICN Information-Centric Netwoking

ICT Information and communications technology

IP Internet Protocol

LS Link-State

NAT Network Address translation

NDN Named Data Networking

PIT Pending Interest Table

QoS Quality of Service

RLRNN Reinforcement Learning with Random Neural Network

RNN Random Neural Network

SANs Self-aware networks

SPs smart packets

Chapter 1

Introduction

1.1 Motivation and Objectives

The success of the Internet is evident especially by its exponential growth, both in the number of communicating components, and its diverse applications. It has become a crucial part of our daily life, so much so that in July 2016, the United Nations passed a resolution, although non-binding, condemning the intentional denial or disruption of access to the Internet as a human right violation. However, this growth has been accompanied by many challenges with managing issues relating to scalability, security, mobility, and Quality of Service (QoS). To tackle these challenges, network engineers have, over the years, proposed intelligent solutions ranging from patches that address specific issues to proposals for a complete re-engineering of the Internet's underlying architecture.

Furthermore, the increasing demand for new and intelligent communication systems with physical realities has led to network technologies evolving from the traditional data-only networks into much more sophisticated systems. Autonomic networks with computational capabilities that can monitor and control physical processes in their immediate environment have become key infrastructures in the modern society. For instance, VigiNet, described in [HKL+06], is a wireless sensor network designed to detect and track moving vehicles which can have useful military applications. Similar systems are also being applied for environment monitoring

[OS12, Meh14], in manufacturing [MKB+16], and in Medicine [LCLZ15]. It is evident that such systems can improve the quality of life of citizens both economically and socially.

To address these issues of managing the successes of the Internet and the evolution of networking technologies, active next-generation networking and future Internet architecture research communities have emerged [DDF+06, PPJ11b, PPJ11a]. These communities explore fresh ideas for the underlying network architecture, context-aware networking, routing and search, mobility, and security.

This thesis focuses on adaptive routing in communication networks and, specifically, the Cognitive Packet Network (CPN) technology [GXS99, Gel09]. CPN is a QoS-driven routing protocol initially proposed as a more dynamic alternative to the conventional Internet Protocol (IP) routing algorithms. It adopts an on-demand approach such that the network nodes obtain network information only when they need it using intelligent exploration, measurement, and adaptation techniques. As a result, CPN avoids network synchronization processes which have limited the dynamism and scalability of the conventional approaches. CPN has been successfully applied in different communication systems, including intercontinental overlay networks [BWG16], Cyber-Physical Systems (CPS) [GSX01, BG14, GB14a], Software-Defined Networks [FG16], Wireless Sensor Networks [Hey08]. We believe that CPN's success is because it allows for scalable decentralised control in communications. And, since networking research is moving rapidly towards autonomic communication, CPN will remain relevant with more opportunities for novel applications. To investigate this hypothesis, this research considers two different but highly dynamic communication systems.

Information-centric networking (ICN) [KCC+07, FP7b, FP7a, NSF, FP7c, FP7d] is now one of the leading research topics in the future Internet architecture research community. ICN proposes replacing the Internet's current host-centric architecture with a content-centric one to match the current demands on the Internet. This approach involves making contents directly addressable at the network level using "names", thereby decoupling content from location. Because of this decoupling, network forwarding entities are able to support in-network caching and replication [XVS+14, ZLZ15] and, therefore, satisfy requests. This thesis tries to answer

the question of how CPN can be used in the ICN context. Specifically, can the CPN's search technique by adapted to effectively exploit the in-network caching capability of ICN-based networks? This is significant because in-network caching can significantly improve content delivery and user experience.

Secondly, emergency navigation systems (EMS) [GGW12] are Cyber-Physical Systems that can alert and intelligently guide both the evacuees and the first responders in confined spaces in an emergency scenario. Exploiting CPN's QoS-driven approach, can the routing system of an EMS be designed to provide different services to the evacuees based on their ongoing health condition during an evacuation? Recognising the needs of different types of evacuees based on their age, mobility, and resistance to fatigue, and tailoring the navigation system accordingly, could improve the survival chances of the evacuees.

To answer these questions, the analysis approach in this thesis is based on simulations using high-fidelity simulators [MAZ17, DFG10] from the relevant literature.

1.2 Contributions

In this thesis, we explored new applications and contexts for CPN. The thesis considers two different but dynamic communication systems. It first addresses the content placement problem in ICN-based networks. Then, a CPN inspired request forwarding algorithm is proposed and evaluated for an ICN architecture. Finally, CPN's QoS capability is exploited in the design of a navigation system for guiding evacuees in confined spaces.

The main contributions of our work are listed as follows.

- We analysed the content placement problem in ICN-based networks as an on-demand problem where the caching decisions follow the user requests. Based on our analysis, we proposed approximate solutions and a cache management framework for implementing these solutions. Our results show that our solution can produce average delay results within 5% of a near-optimal one.

- We proposed a Reinforcement learning algorithm using the Random Neural Network, inspired by CPN, for request forwarding in named data networks. By using a delay metric, our proposed algorithm proactively searches effectively for local cache hits. Furthermore, we introduced control measures that guarantee that our proposed strategy is correct if the routing table has been constructed without loops. Our results show that our proposed strategy can exploit in-network caching better than a strategy that restricts its search to the paths in the routing table and a strategy that explores an entire network.

- We proposed a dynamic grouping mechanism based on the ongoing health conditions of evacuees during emergency navigation in confined spaces. In other words, evacuees can change their groups during an emergency evacuation according to their current health conditions. CPN routing is then tailored to find multiple paths and provide different levels of services to the evacuees based on their assigned group. Our evaluations show promising results which suggest that using different path metrics for different evacuees and the dynamic grouping mechanism can improve the survival chances in an evacuation.

1.3 Publications

Parts of this thesis have been presented and published in the following peer-reviewed conferences and journals:

- O. Akinwande, and H. Bi. "Routing diverse crowds in emergency with dynamic grouping." In Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on, pp. 487-492. IEEE, 2015. [AB15].

- H. Bi and O. Akinwande. "Multi-path routing in emergency with health-aware classification." Information and Communication Technologies for Disaster Management (ICT-DM), 2015 2nd International Conference on, pp. 109-115. IEEE, 2015. [BA15].

- H. Bi, O. Akinwande, and E. Gelenbe. "Emergency Navigation in Confined Spaces Using Dynamic Grouping." In Next Generation Mobile Applications, Services and Technologies,

2015 9th International Conference on, pp. 120-125. IEEE, 2015. [BAG15].

- O. Akinwande, H. Bi, and E. Gelenbe. "Managing crowds in hazards with dynamic grouping." Access, IEEE 3 (2015): 1060-1070, [ABG15].
- O. Akinwande and E. Gelenbe. "A Reinforcement Learning Approach to Adaptive Forwarding in Named Data Networking." In Proceedings of the 32nd International Symposium on Computer and Information Sciences, pp. 211-219. Springer International Publishing, 2018, [AG18].
- O. Akinwande. "Interest Forwarding in Named Data Networking Using Reinforcement Learning." Sensors (Basel, Switzerland) vol. 18,10 3354. 8 Oct. 2018, [Aki18].

1.4 Thesis Outline

In Chapter 2 we briefly outline the research in Autonomic Communication, Information-Centric Networking and Cyber-Physical Systems.

In Chapter 3, we present the Cognitive Packet Network architecture. We outline the previous applications of CPN in both wired and wireless networks, and we detail the learning algorithm used in CPN for routing and search.

In Chapter 4, we describe the Named Data Networking architecture, one of the prominent ICN projects, which we have adopted for our evaluations. We also outline the models and assumptions for the information-centric networks we evaluate.

In Chapter 5, we analyse the content placement problem in ICN and present approximate solutions and a framework for implementing adaptive caching decisions in an ICN-based network.

In Chapter 6, a request forwarding strategy, inspired by CPN, is presented for the NDN architecture to exploit in-network caching.

In Chapter 7, we present a routing algorithm, inspired by CPN, for directing evacuees in confined spaces during an emergency.

Finally, Chapter 8 provides a summary of the results and the future research directions.

Chapter 2

Background

2.1 Networked Self-Organised System

Autonomic communication [DDF+06] is the development of self-managing networks. One of the primary motivations for this research direction is the increasing demand for networks that can self-configure, self-adapt, self-optimise and self-heal. For instance, a wireless sensor network deployed for forest fire detection should operate successfully with minimum manual intervention. Furthermore, autonomic communication is being proposed as a better way to manage the complexities resulting from the continuous growth of the Internet by replacing centralised control infrastructures with decentralised communication.

The design of autonomic networks is usually characterised by decentralised algorithms, learning algorithms, algorithms mimicking the behaviour of natural systems [DCD98, DC+04, SDC97, Cos02, WFZ04] and algorithms based on game theory. This shift towards a decentralised and non-deterministic approach means that for autonomic communication, fresh methods are required to address the accompanying reliabilty, stability, correctness and security challenges.

In this research, we exploit methods in the Cognitive Packet Network (CPN) [GXS99], one of the prominent contributions in the autonomic communication research, in two different types of highly dynamic networked systems for which traditional networking approaches are

not scalable. Specifically, we adopt the search algorithm in CPN, which uses Reinforcement Learning with Random Neural Network (RLRNN), to two different networks to optimise well-defined performance metrics.

2.2 Information-Centric Networking

The Internet is currently built on a model designed to tackle the resource sharing problem encountered early on in data communication where one computer seeks to access resources on another remote computer. Specifically, there was a limited number of powerful computers for research and the many researchers that required access where geographically separated from them. The result was a model where communication is between two fixed machines whose identities are known to the network in every exchanged message.

When computers became small enough to become mobile, adjustments were necessary to handle mobile communication. Mobile IP [Per10] was introduced to provide a workaround for mobile nodes to send and receive Internet packets while still using the Internet's addressing scheme which is designed and optimised for stationary hosts. This workaround comes at the cost of introducing inefficiency in packet forwarding, additional infrastructure, and increased complexity [PJA11, AVH07]. Virtualisation and tunnelling techniques have helped to introduce some level of privacy by hiding the true identities of the communicating nodes from the Internet. These techniques have also allowed the development of technologies like the Network Address translation (NAT) and overlay networks. NAT allows devices with private addresses to communicate on the global Internet, helping to conserve the address space and provide an additional layer of network security. Overlay networks create arbitrary topologies on top of the Internet topology which is useful in enforcing custom routing policies that offer QoS guarantees [BCS94, BBC+98] or improve routing generally [BWG16]. Authentication combined with encryption help to secure two ends of a connection and the messages exchanged [SK05], and firewalls control a networks access to the global Internet. Furthermore, the IP multicast [CDF+02] is a solution for one-to-many and many-to-many communication, while proxy servers, Con-

tent Delivery Networks (CDNs) and P2P systems are solutions that make the Internet appear smaller by bringing services and contents closer to users.

On the other hand, rather than developing more patches, there is an active research community arguing for replacing the current Internet architecture with one that better matches it with its current demands while being flexible enough to handle future requirements [PPJ11b, PPJ11a]. An approach proposed by this community that is of interest to us and is becoming increasingly popular is Information-Centric Netwoking (ICN). The fundamental concept in ICN is to make content directly addressable which is a significant departure from the host-centric communication model of the current Internet. To achieve this, ICN names and addresses data units at the network level, so that in the place of a destination address, request packets are routed and satisfied based on the name or identification of the desired content.

The main motivation for this approach is the increasing domination of Internet traffic by multimedia, effectively making the Internet a content distribution network. According to Cisco's latest forecast in [Cis17], 82 percent of all consumer Internet traffic will be video traffic by 2021 and 71 percent of all Internet traffic will cross CDNs by the same year. Despite the success of CDNs, they can become limited and complicated because the underlying host-to-host communication infrastructure is agnostic of content [WF11]. By treating data as fundamental to the architecture, ICN seeks to match these demands placed on the Internet and improve on the patch solutions. The separation of content from location enables ICN to support desirable features which facilitate efficient content delivery like Internet-wide in-network caching and replication, and multicast communication. In addition to treating content as fundamental to the architecture, ICN also addresses mobility and security challenges of the Internet.

It is important to mention that the ideas for a content-centric network were first introduced in the TRIAD project [CG00a] almost 2 decades ago, motivated by the scaling issues of meeting the increasing demand for content over the Internet. These ideas were developed further in [GC01] as a better alternative, in terms of the access time to content servers, to the conventional DNS-based systems for solving the content routing problem of the Internet. This involved integrating content routing into the network via a *content layer*. This layer is implemented by specialised

routers, called *content routers*, and a name-based routing protocol. The content routers direct a name lookup request towards the content router closest to the preferred content server, and the address of the server is returned to the requesting application. The role of the name-based routing protocol is to dynamically maintain the name to next-hop mappings at the content routers. Also, the works in [CW02, CRW04, CW03] explored content-based networks to realise a receiver-driven style of communication which is suitable for information sharing systems and mobile/wireless computing. In their content-based network, proposed as an overlay, each receiver/user advertise the contents of interest to it as a content-based address. The network then provides the service of delivering theses contents by disseminating the addresses to all possible sources.

Currently, several projects are developing the ICN concept, including DONA [KCC+07], CONVERGENCE [FP7b], COMET [FP7a], Named Data Networking (NDN) [NSF], PURSUIT [FP7c] and SAIL [FP7d]. ICN also presents fresh challenges in the naming of data objects, cache management policies, routing of data objects and requests, and in the handling of mobility and security. While the ICN projects all share the same central idea, they differ significantly in how they address these challenges [XVS+14, ADI+12]. Of most significance to us in this thesis are the cache management and the forwarding of request problems.

In-network caching allows the network forwarding entities the capability of caching data objects and satisfying the requests that match the contents in their caches. This will present fresh challenges and opportunities to network administrators, as it will be in their interest, both in terms of customer experience and transit costs, to serve as many requests as possible locally. Also, administrators could offer publishers special services whereby their contents receive priority in the cache allocations. Furthermore, to exploit in-network caching, some ICN projects have adopted a stateful forwarding plane in their architectures. This presents a platform for exploiting learning algorithms and adaptation to optimise delivery performance. In this thesis, we address the exploitation of in-network caching in ICN by developing a dynamic system for managing the caching decisions in a network of caches and an adaptive strategy for forwarding of request packets.

2.3 Cyber-Physical Systems and Motion in Confined Spaces

Cyber-Physical Systems (CPS) are computational systems with physical realities. CPS comprises of cooperating computational entities which monitor and control physical processes in their environment. In other words, CPS are usually sensor-based autonomous systems with communication capabilities.

As a result of the advancements in Information and communications technology (ICT), cyber-physical systems have become crucial in improving the quality of life of citizens and the competitiveness of several industries in modern societies.

For smart rodent control, a CPS was proposed in [Meh14] for rat detection using motion detectors and thermal cameras. Such a solution can be used to reduce the damage to crops and be a substitute for pesticides which damage the environment. Kantarci [Kan15] proposed an intelligent transport system, using vehicular communications, for computing and recommending alternate routes for paramedics in urban areas. CPS have also been applied in Medicine, for instance, to improve human error detection in medical environments [LCLZ15].

Of particular interest in this research are cyber-physical systems for directing motion in confined spaces. Such systems can be useful in guiding both the evacuees and the first responders in confined spaces during emergency situations, thereby significantly improving the survival chances of the evacuees and reducing the damage to properties.

In [LDRR03], distributed algorithms are proposed which use the danger levels measured by a sensor network to compute safe paths for evacuees away from the danger areas. In our work, we tackle a more difficult problem and present a quality of service (QoS) driven routing algorithm that caters to the needs of different types of evacuees based on age, mobility, and level of resistance to fatigue.

Chapter 3

The Cognitive Packet Network

3.1 Introduction

The Cognitive Packet Network (CPN) is one of the significant contributions to Self-aware networks (SANs) [Gel09, GLN04]. In SANs, a network node can autonomously discover other nodes, links, and paths whenever it desires such information irrespective of the network topology. SANs achieve this, in addition to performance optimisations, through exploration, measurement, and adaptation. SANs represent a departure from traditional network designs where dynamism is usually traded to maintain constant connections between the nodes in the network.

The CPN approach uses three types of packets: smart packets (SPs), acknowledgement packets (ACKs) and dumb packets (DPs). SPs are sent for network exploration to establish and possibly improve the connection between two nodes. They record useful measurements about the network and are routed using the learning algorithms residing in each network node. When an SP arrives at its intended destination, an ACK is returned to the SP's source carrying the path information discovered by the SP. DPs, on the other hand, carry actual data and exploit the information discovered by the SPs.

One of the main motivations for developing CPN was to address the lack of frameworks for exploiting learning algorithms and adaptation in communication in computer networks. The

CPN architecture allows for decentralised control in communication and for realizing autonomic networks that are self-managing, self-configuring and self-regulating.

The CPN routing algorithm was first described in a conference paper in 1999 [GXS99], where the design of the network's routing engine exploited the function approximation properties of the Gelenbe's Random Neural Network (RNN) [Gel89, GML99] to obtain a reliable and predictable improvement in the QoS experienced by end users, based on the QoS criteria requested by the end users themselves. The first experimental results regarding CPN were presented in [GLX01b, GLX01a, GLMX02], while a US patent for CPN was awarded in 2004 [Gel04b]. Simultaneously, the CPN routing algorithm was implemented in an ad-hoc wireless network to obtain savings in energy consumption [GL04] and pursued several years later also for wired networks [GM11].

Early on, the CPN idea was tested in a specific simulation context [GSX01] to see how it could be used to help vehicles avoid being destroyed by enemy fire in the context of urban warfare. Interestingly enough, the idea was later picked up in a related context to guide evacuees during emergency evacuations [BG14, GB14a] to maximise the chances of survival for human evacuees.

Since CPN uses Reinforcement Learning with a Random Neural Network for decision making [Gel89, Gel93c, GH02], it was presented as an innovative application of neural information processing [Gel04a]. Later, CPN was extended to include the search for alternate network paths using genetic algorithms [GLL06]. Early work considered the optimization of the QoS for a source to destination connection using CPN, while more recent work has recognised and found solutions to the challenges of connections that have simultaneous but differing requirements for the source to the destination, and the destination to the source [GK14], as with a user's connection to a web site from which it downloads media content. Another area that has received attention concerns the use of CPN for the specific QoS needs of voice traffic [WG15a] as in Skype or similar systems.

The early 2000's were also a time when Autonomic Computing was being launched by industry and notably by IBM so that CPN became a constituent of the new movement for Autonomic Communications [GGL+04, DDF+06]. A new research area on *Self-Aware Networks* [GLN04,

Gel09] was also developed, and it rapidly gave rise to interesting applications regarding the defense of a network against Distributed Denial of Service attacks [GL07].

While CPN was initially designed as a way for providing a routing substrate that replaces IP networks, recent work [BWG16] has shown that it can be used to substantially improve QoS for intercontinental overlay networks as well. Further work has been conducted to design cloud server platforms that use CPN to allocate tasks to different servers for the diverse QoS needs of different types of compute bound or I/O bound or other kinds of tasks [WG18, WG15b, GW16], and this work has been extended to deal with remote clouds, where both the network QoS and the actual cloud server QoS is taken into consideration [WBG16]. Finally, let us mention that CPN has also been used and tested to automate the set-up of Software Defined Network connections [FG16].

This chapter outlines the CPN architecture and its applications in routing in wired and wireless networks. The chapter is organised as follows. In Section 3.2, a review of the research in adaptive routing in computer networks is presented. Section 3.3 presents a reinforcement learning using the RNN algorithm which commonly used in CPN routing. Section 3.4 examines goal function-based routing using CPN, and Section 3.5 explains how CPN is adapted for wireless networking. The chapter concludes in Section 3.6.

3.2 Adaptive Routing in Networks

The two classes of the traditional routing algorithms for computer networks are the Distance Vector (DV) algorithms [MW77, MFR78, Hed88, HB91, RLH06] and the Link-State (LS) algorithms [MRR80, KZ89, Moy97, Ora90]. These algorithms employ already established solutions for computing shortest paths in graph theory. The DV algorithms are based on the distributed version of the original Bellman-Ford algorithm [BG92], while the LS algorithms commonly use Dijkstra's algorithm [Dij59] for path computation.

DV algorithms have very well-known convergence problems because of the distributed nature of updating their routing tables when there is a change in the state of the network [Med10, PD07].

On the contrary, LS algorithms adopt a centralised approach in which each node in the network maintains a copy of the complete network topology from which it computes the shortest path tree to every other destination. In LS algorithms, each node periodically floods the network with updates of the costs of its outgoing links. As a result, LS algorithms possess better convergence properties compared to the DV algorithms, but they incur much more computation and communication overhead.

However, LS algorithms can still suffer from convergence issues because of inconsistent information at the nodes. The problem of maintaining correct and identical global routing information at every node is known as the *routing information problem* (RIP) [Spi85] or the *distributed database problem* [Ros80]. This is not a trivial problem especially in large networks that experience regular changes in its topology. For example, in the Open Shortest Path First (OSPF) protocol, the most commonly used LS algorithm in large Enterprise networks, to better manage the routing information problem and scalability, static link costs that are only reconfigurable by a network administrator are used. Also, the interval between the broadcast of link state updates is as high as half an hour; only link failures trigger the update process outside this interval. It is easy to conclude that high-level dynamism is sacrificed for scalability in many large networks.

Adaptive routing algorithms have been studied to address this lack of dynamism in computer networks because of the increasing importance of mobile ad-hoc networks and increasing domination of Internet traffic by media [Gel03, CN98]. Q-routing [BL94], which was derived from the Q-learning algorithm, is the first algorithm to apply Reinforcement learning in packet routing. In Q-routing, there is an initial exploration phase where nodes make random decisions when forwarding packets. With every decision a node makes, its learning algorithm updates a Q value corresponding to that decision. Eventually, the algorithm should converge to a routing policy which is expected to give the best path between two nodes. The routing policy at a node for a given destination is to choose the next-hop node with the optimal Q value. Modifications were later introduced to Q-routing to improve its exploration and convergence to the best policy [CY96, KY05, OJ10, KM97].

Agent-based routing algorithms [DCD98, DC$^+$04, SDC97, HSGK98, WFZ04] are a class of adaptive routing algorithms that have enjoyed considerable attention from researchers. In these algorithms, a mobile agent traverses the network to learn the network topology towards establishing and improving connections between the nodes in the network. The agents either "experience" the network conditions themselves like an actual data packet or they learn from local estimates at the nodes they traverse. Data packets exploit the information gathered by the agents. In many agent-based algorithms, the movement of the mobile agents imitate properties of social insects, like ants [DCD98, DC$^+$04, SDC97, Cos02] and bees [WFZ04], when they perform complex tasks like finding the best path to a food source.

Genetic algorithms which code evolutionary operations of selection, crossover, and mutation based on the work of Charles Darwin in biology, to solve optimisation problems [Gol89], have also found application in network routing. In a routing algorithm, a path between two nodes can represent a chromosome or an individual, and its fitness is the cost of sending a packet along it. The genetic operators are used to combine known paths to generate more network paths that can be explored. Any routing algorithm that can compute the sequence of nodes along network paths can easily apply genetic algorithms.

In the *Genetic Adaptive Routing Algorithm* (GARA) [Mun97, MTS98], each node maintains full topological information and initial paths are generated using the Dijkstra's algorithm as in the LS methods. Genetic operators are then applied at specific intervals to find more paths, and special packets, called *delay query packets*, are sent along the discovered paths to evaluate their costs. Nodes also share their discovered routes with neighbouring nodes at regular intervals. Genetic algorithms have also been applied in agent-based routing strategies [CH03, LZHH02].

The *Cognitive Packet Network* (CPN) [GXS99] uses a QoS-driven routing protocol whose approach is similar to the agent-based algorithms. In CPN, nodes obtain network information only when they need it, thereby eliminating the need for network-wide synchronization processes as in the traditional routing algorithms. Exploration packets, called smart packets (SPs), are sent to gather relevant network information necessary for path discovery. In most implementations, the SPs are guided to the most worthwhile regions of a network using the random neural net-

work (RNN) [Gel89]. CPN has also been extended to include the search for alternative network paths using genetic algorithms [GLL06].

3.3 Reinforcement Learning and the Random Neural Network

The RNN, developed by Gelenbe [Gel89, Gel90a, GF99, GT08], is a neural network model inspired by queueing networks [GP98] where neurons send and receive both positive and negative signals. The positive signals are called *excitatory*, while the negative signals are called *inhibitory*. The RNN has been shown to possess function approximation properties [GML99, GhMdL14]. Furthermore, it has been used in many applications, including image and video compression [GSCG96, CG00b, CGB96], the recognition of tumours from Magnetic Resonance Images of the human brain [GFK96], toxicity prediction of chemical compounds [GYCG18], developing heuristics for classical combinatorial optimisation problems [GB92, GKP93], web search [SG18], and network routing and cloud management [GGS97, Gel04b, GL04, GLL06, FG18, WG18]. The RNN is a special instance of the family of stochastic networks known as G-Networks [Gel91, GGS91, GS92, Gel93a, Gel93b, FG17] which have many different areas of application.

Table 3.1 lists the notation for the RNN model.

Table 3.1: Notation for the RNN model.

Notation	**Definition**
$k(t)$	State vector of the RNN at time t where $k_i(t) >= 0$ is the potential level of a neuron i in the network
r_i	Firing rate at neuron i
Λ_i	Arrival rate of positive exogenous signals at neuron i
λ_i	Arrival rate of negative exogenous signals at neuron i
$p^+(i,j)$	Probability that a signal leaving neuron i is excitatory and heads for neuron j
$p^-(i,j)$	Probability that a signal leaving neuron i is inhibitory and heads for neuron j
$d(i)$	Probability that a signal leaving neuron i departs the network

The state of a neuron i in the network at any time is represented by a non-negative potential value $k_i(t)$, so that the vector:

$$k(t) = (k_1(t), k_2(t), ..., k_n(t))$$

describes the state of an RNN at any time t. An incoming signal either increases a neuron's potential by one if it is excitatory or decreases it by the same amount if it is inhibitory. An excited neuron i, that is, one with non-zero potential ($k_i(t) > 0$), fires randomly by sending out excitatory or inhibitory signals to other neurons or to the outside of the network. Whenever a neuron fires, its potential reduces by one. Signals can also arrive at a neuron from outside the RNN. Therefore, for each neuron i in an n-neuron network, we can write

$$\sum_{j}^{n}[p^+(i,j) + p^-(i,j)] + d(i) = 1. \tag{3.1}$$

The weight of each RNN connection (i, j) is the emission rate of positive or negative signals from neuron i to neuron j. That is,

$$w^+(i,j) = r_i p^+(i,j), \quad w^-(i,j) = r_i p^-(i,j)$$

Figure 3.1 shows a simple representation of the RNN model.

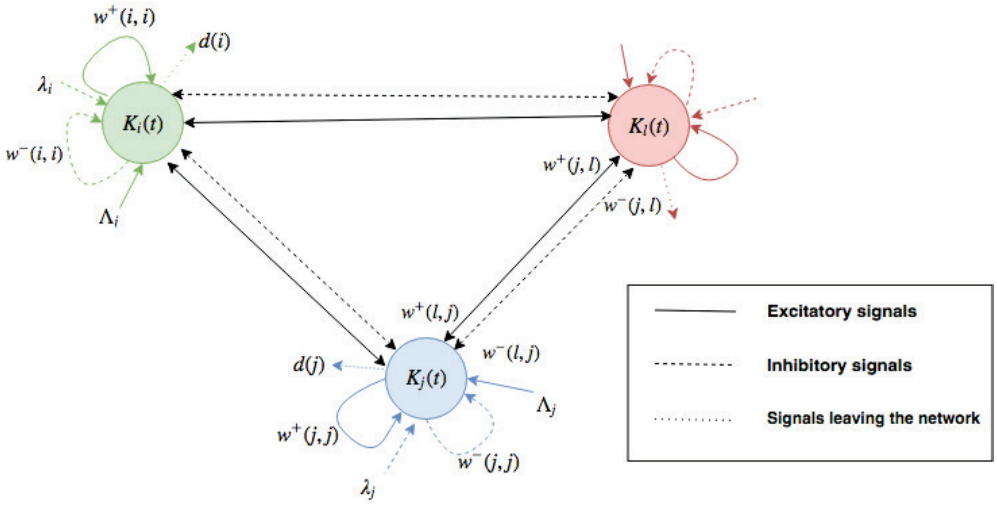


Figure 3.1: A simple representation of the RNN model.

It was shown in [Gel89] that the RNN with exponential firing intervals and Poisson exogenous signal arrivals has the product form solution:

$$p(k) = \prod_{i=1}^{n}(1 - q_i)q_i^{k_i},\ q_i = \frac{\lambda_i^+}{(r_i + \lambda_i^-)}, \tag{3.2}$$

where $p(k) = \lim_{t\to\infty} Pr[k(t) = k]$, $k = (k_1, k_2, ..., k_n)$, is the network's stationary probability distribution; q_i is the steady state probability that neuron i is excited. λ_i^+ and λ_i^- represent the overall flow of positive and negative signals, respectively, at neuron i. For $i = 1, 2, ..., n$, λ_i^+ and λ_i^- satisfy the following system of non-linear equations whose solution exists and is unique [Gel93c]:

$$\lambda_i^+ = \sum_j [q_j r_j p^+(j, i)] + \Lambda_i \tag{3.3}$$

$$\lambda_i^- = \sum_j [q_j r_j p^-(j, i)] + \lambda_i \tag{3.4}$$

Assuming signals do not leave the network ($d(i) = 0$), we can rewrite equation (3.1) as

$$r_i = \sum_{i=1}^{n} [w^+(i, j) + w^-(i, j)] \quad i = 1, 2, ..., n \tag{3.5}$$

The matrices of the weights of the connections $W^+ = \{w^+(i, j)\},\ W^- = \{w^-(i, j)\}$, are important parameters whose values are continuously modified during a learning process.

3.3.1 CPN Routing

The learning algorithm used in CPN is a RLRNN algorithm. The algorithm attempts to direct SPs to the most worthwhile areas of a network in their search for network paths based on a well-defined metric.

In CPN, nodes obtain network information only when they need it, eliminating the need for any form of network-wide synchronization as required in most traditional routing algorithms. It uses three types of packets: *smart packets* (SPs), *acknowledgement packets* (ACKs) and *dumb packets* (DPs).

A node requiring a connection will send out some amount of SPs towards the destination node. The RLRNN algorithm is used at each router for routing the SPs. An RNN at a router has the same number of neurons as the number of outgoing links or forwarding options available at the router. And, it is uniquely identified by the destination it points to and its QoS class.

When an SP arrives at a node, it records relevant routing information based on the chosen path metric, including the identity of the node. If the arrival node has no information about the SP's destination, that is, either no RNN exists matching the desired connection or the RNN is still in its initial state without any feedback from its previous decisions, a random decision is made in choosing the SP's next hop. Otherwise, with a high probability d close to 1, the RNNRL algorithm selects the interface corresponding to the RNN's most excited neuron to send the SP or a random decision is made with probability $1-d$. This randomization to ensure continuous exploration of the network. In many of the CPN literature, d is referred to as the *drify parameter*. Note that if no matching RNN exists, the arrival of an SP triggers the creation of one.

When an SP arrives at its destination, an ACK packet is generated by the destination node. The ACK carries back to the source node, the measurements recorded in the SP along the "reverse path" used by the SP after all loops have been removed. The ACK provides the source node with full path information for the requested connection. Also, ACKs update the corresponding RNNs at the nodes they traverse, and this triggers the RNNs to adjust their weights.

A DP, on the other hand, carries the actual data or payload and uses the highest ranked path information at the source node to route itself. ACKs are also generated for DPs when they arrive at their destinations which means DPs can also be used to monitor the condition of a connection. The DPs and ACKs are source-routed in CPN. After establishing a connection, subsequent SPs are sent by the source at a reduced rate to maintain and, if possible, improve the connection.

The pseudocode in Algorithm 1 illustrates how the RLRNN algorithm updates its weights when it receives a feedback from an ACK. This algorithm assumes that a reward value can be estimated for every forwarding decision. In this approach, the outcome of a previous decision,

R_l, is compared to an internal expectation, T_{l-1}, of the neural network, and the RNN receives reinforcement based on the result of this comparison. The reinforcement can either be *positive* if the outcome is considered a success, or *negative* otherwise. In the illustration in Algorithm 1, the l-th decision is considered to be a success if $R_l \geq T_{l-1}$; otherwise it is a failure. A success leads to a significant increase in the excitatory weights going into the corresponding neuron and a small increase in the inhibitory weights leading to the other neurons, as shown in lines $3-6$. Otherwise, the inhibitory weights of the neuron assigned to the decision is significantly increased and the excitatory weights into the remaining neurons are slightly increased, as shown in lines $9-12$.

Algorithm 1 The RNN training process

INPUT: T_{l-1}, W^+, W^-, internal expectation and weight matrices before the l-th reward is received ; $0 < \alpha < 1$.

INPUT: ${R^i}_l$, the l-th received reward obtained when the lth decision was to select output link i.

OUTPUT: q_i, $\forall\ i = 1, 2, ..., n$, the excitation probabilities for an RNN with n neurons

1: **if** ${R^i}_l >= T_{l-1}$ **then**
2: /* l-th decision is a success*/
3: **for** each neuron j in the RNN, $j \neq i$ **do**
4: $w^+(j,i) \leftarrow w^+(j,i) + {R^i}_l$
5: **for** each neuron k in the RNN, $k \neq j$ **do**
6: $w^-(k,j) \leftarrow w^-(k,j) + \frac{{R^i}_l}{n-2}$
7: **else**
8: /* l-th decision is not a success*/
9: **for** each neuron j in the RNN, $j \neq i$ **do**
10: $w^-(j,i) \leftarrow w^-(j,i) + {R^i}_l$
11: **for** each neuron k in the RNN, $k \neq j$ **do**
12: $w^+(k,j) \leftarrow w^+(k,j) + \frac{{R^i}_l}{n-2}$
13: /* Normalizing the weights */
14: **for** each neuron j in the RNN **do**
15: $r_j^* \leftarrow \sum_m^n [w^+(j,m) + w^-(j,m)]$
16: **for** each neuron k in the RNN, $k \neq j$ **do**
17: $w^+(j,k) \leftarrow w^+(j,k) * \frac{r_j}{r_j^*}$
18: $w^-(j,k) \leftarrow w^-(j,k) * \frac{r_j}{r_j^*}$
19: /*solve the n non-linear simultaneous equations in (3.2) for each $0 \leq q_j \leq 1$ */
20: /* We use a fixed-point iteration, starting with $q_j^0 = 0.5\ \forall\ j = 1, 2, ..., n$ */
21: $q_j^{k+1} \leftarrow min[1, \frac{\sum_m [q_m^k\ w^+(m,j)] + \Lambda_j}{r_j + \sum_m [q_m^k\ w^-(m,j)] + \lambda_j}]$
22: /* Updating the internal expectation */
23: $T_l \leftarrow \alpha T_{l-1} + (1-\alpha) R_l$ /* α is to be chosen closer to 1 */

Afterwards, the weights are normalised as shown in lines 17 and 18. This is to prevent ever-

increasing weights, and the fact that it is the relative magnitudes of the weights that determine the state of the network justifies this step. Before normalising, a new firing rate r_i^* for each neuron is first evaluated using Equation (3.5). Then, for each neuron i the normalised weights are computed as

$$w^+(i,j) = w^+(i,j) * \frac{r_i}{r_i^*},$$
$$w^-(i,j) = w^-(i,j) * \frac{r_i}{r_i^*},\ j = 1, 2, ..., n,$$

where r_i is the initial or reference firing rate.

Finally, the excitation probabilities q_i for all the neurons are evaluated by solving the non-linear system of equations

$$q_i = \frac{\lambda_i^+}{(r_i + \lambda_i^-)},$$
$$\lambda_i^+ = \sum_j [q_j r_j p^+(j,i)] + \Lambda_i,$$
$$\lambda_i^- = \sum_j [q_j r_j p^-(j,i)] + \lambda_i.$$

The common approach to solving these equations is using the simple fixed-point iteration as shown in line 21. At any given time, the neuron with the highest excitation probability q is considered the best decision. Finally, the internal expection is updated with the new reward using

$$T_l = \alpha T_{l-1} + (1 - \alpha) R_l,$$

where $0 < \alpha < 1$.

This algorithm has a computational complexity of $\mathcal{O}(n^2 I_k)$, where n is the number of neurons in the RNN and I_k is the number of iterations used in computing the excitation probabilities. The current CPN testbed [Len03] adopts the following input values in its implementation: $\Lambda_i = 0.125$, $\lambda_i = 0$ for $i = 1, 2, ..., n$; and the initial value of all the weights is set at 0.5.

3.3.2 Convergence of CPN Routing

The convergence of the reward/punishment learning scheme for RNN based on an internal expectation referred to as the *E-rule* scheme, was first analysed in [Hal00]. The simulation results in [Hal00] also showed that the E-rule scheme outperforms a reward only scheme. Also, the

E-rule scheme is sensitive to environmental changes because it promotes exploration. Desmet and Gelenbe in [DG14] studied the influence of the drift parameter on convergence in CPN routing. Although, their simulations were in the context of a building evacuation where each node in the network initiates, at the same time, a connection to the same sink or exit nodes. As a result, nodes can benefit from the abundance of information since their SPs share similar goals. Therefore, in this scenario, the choice of the drift parameter value represents a tradeoff between path discovery time and the speed of convergence to the optimal path.

Since one of the primary requirements of CPN is to establish connections on demand, it is important to study how effective CPN is in finding paths through a random walk or search. A completely random search occurs when no RNN in the network has prior information about the desired connection. Also worth studying, is the rate at which the source of a connection should send subsequent SPs after a connection has been established to ensure that the defined QoS for the connection is reasonably achieved.

These subjects were studied in [GLX01a, GLN04] via extensive experimental works. In [GLN04], a network topology with an infinite number of SP paths between any 2 nodes was considered. Since SPs are allowed to "loop", such a network can easily be constructed by having nodes with high degrees or number of connections. Their results showed that by setting a maximum hop for the SPs, only a small number of SPs is necessary for the initial path setup. Furthermore, Gelenbe et. al. in [GLX01a] considered a stream of traffic between 2 nodes and showed that a comparatively small fraction of SPs (about 15 -20% of the entire user traffic) achieved a substantial gain in QoS.

While [GLN04] experimented with a random walk by the SPs, in applications were overhead is not a major concern, SPs can be flooded at nodes with no prior information. Another approach is to allow network nodes to access the full path already travelled by an SP to prevent it from looping. Liu and Gelenbe in [LG07] proposed recursive routing in CPN to address large-scale routing problems. In their approach, CPN nodes can store the paths in the ACKs they receive; and, an SP discovers a path by reaching either its intended destination or a node with a suitable path to the intended destination. Their results showed a significant reduction in the

time required to establish connections and improved QoS.

3.4 Goal Function-based Routing in Networks

CPN was designed to facilitate QoS-based routing using adaptive techniques. The role of the SPs is to explore a network and estimate as best as possible the quality of the paths they discover. As a result of the feedback mechanism provided by the ACKs, network nodes are able to compute "reward" values, R, which quantify their forwarding decisions. The RNNRL algorithm then updates its weight depending on whether R is a success or a failure.

To implement CPN routing, a goal function, G, is defined for each QoS class in the network. The goal function defines the path metric that the SPs belonging to the corresponding QoS class must pursue in their exploration. For instance, a goal can be expressed directly as minimising delay (D), minimising number of hops (H), minimising loss (L), minimising jitter (J), or a weighted combination like minimising both delay and loss. Since QoS goals generally require a minimisation, R is commonly computed as

$$R = \frac{1}{G} \tag{3.6}$$

Delay is the default QoS goal implemented on the CPN testbed. An SP records the arrival time at each node it visits so that the forwarding delay D_t at time t when its corresponding ACK arrives at a node is estimated as

$$D_t = \frac{t - t_{sp}}{2}, \tag{3.7}$$

where t_{sp} is the arrival time of the SP. The delay goal G_t^D is then updated at time t as the smoothed exponential average of the forward delay estimates. That is,

$$G_t^D = aD_t + (1-a)G_{t-1}^D, \quad 0 < a < 1. \tag{3.8}$$

Gelenbe and Liu [GL05] experimented with a hop count metric to adaptively accomplish shortest path routing. The goal, G_t^H is updated for each forwarding decision using

$$G_t^H = \beta * H, \tag{3.9}$$

where H is the number of hops travelled by the ACK and β is a constant set to 0.75.

A composite goal function, G^{DL}, for a QoS requirement including both delay and packet loss was presented in [GGS03] as

$$G_t^{DL} = G_t^D + G_t^L, \tag{3.10}$$

where G_t^L, the loss QoS goal at time is estimated using

$$G_t^L = T\frac{L_t}{1 - L_t}, \tag{3.11}$$

where T is the additional time incurred before a lost packet is retransmitted, and L_t is the smoothed exponential average of the estimates of the forward packet loss ratio of a path. To estimate the forward packet loss ratio of a path, it is assumed that the DP loss ratio along the path is equal to the ACK loss ratio along the "reverse" path. Therefore, at a given time, the forward packet loss ratio is estimated as $1 - \sqrt{\frac{A}{N}}$, where N is the total number of DPs sent along the path, and A is the corresponding number of ACKs received via the path at time t.

To fulfill the tight QoS constraints for real-time transport, Gelenbe et. al. in [GLMX02] implement a composite function, G^{DJ}, that combines both delay and jitter goals given as

$$G_t^{DJ} = G_t^D + G_t^J, \tag{3.12}$$

where G_t^J, the jitter QoS goal at time is estimated using

$$G_t^J = G_{t-1}^J + (|D_t - G_t^D| - G_{t-1}^J)/16. \tag{3.13}$$

Equation 3.13 is the formula for computing interarrival jitter suggested in [FCJS96].

Finally, energy awareness can also be introduced in network routing. In fact, the efficient use of energy in communication networks is a major priority in modern networks and has been extensively studied. A popular approach is switching idle network devices off or switching them to sleep mode under lower transmission demands to minimise power consumption while maintaining communication over the network. Gelenbe and Mahmoodi [GM11] developed an energy aware routing protocol (EARP) for wired networks using CPN routing is presented in [GM11] EARP uses a composite cost which combines power consumption and the expected QoS goals and tries to use paths that minimise the total power consumption in the network while respecting the QoS of individual flows.

3.5 CPN Routing for Ad-Hoc Networks

CPN has also been adapted for wireless networking, especially mobile ad hoc networks (MANETs) [GL04]. A MANET is a wireless network comprising of mobile nodes capable of communicating with each other without relying on any centralised infrastructure. These networks are decentralised, and each node acts both as an endpoint and a relay node for other communications. Furthermore, MANETs are highly dynamic networks and are usually expected to be autonomic, that is, self-managing, self-configuring and self-regulating. Applications of MANET include sensor networks, vehicular communication, disaster recovery, and search and rescue operations. Like other wireless networks, the efficient use of energy is also vital in MANETs.

Gelenbe and Lent in [GL04] proposed the energy sensitive Ad hoc CPN (AHCPN) protocol which searches for routes in a MANET by considering both the energy stored in the nodes and path delay. Due to the high rate of topological changes, SPs in AHCPN may be broadcasted at nodes with no forwarding information for the intended destination. Also, the random exploration of paths based on the drift parameter is also replaced by SP broadcast. Furthermore, SPs are discarded when they travel in a loop, unlike the original algorithm.

In addition to energy awareness, QoS can also be a critical requirement in MANETs. In [GN08, HG09] a routing algorithm for sensor networks is presented which is able to detect and

provide a better level of service to packets that originate as a result of "high priority" events. Priority of traffic is indicated in the packets and once detected, regular packets are shifted to secondary paths to improve the QoS of the high priority packets.

3.6 Summary

In this chapter, an overview of the CPN architecture is presented, focusing on its routing applications in both wired and wireless networks. The chapter first presents a summary of the research relating to CPN. This was followed by a review of the research in adaptive routing in computer networks. Finally, we outlined the learning algorithm commonly used for CPN routing and presented some applications of CPN routing.

Chapter 4

Information-Centric Networks

4.1 Introduction

Information-centric networking (ICN) has become one of the leading topics in the next-generation networking and future Internet architecture research. This is because its underlying model is designed to be a better match to the current demands placed on the Internet compared with the present host-centric architecture.

Since the commercialisation of the Internet, its predominant use has continued to shift towards information dissemination, with users increasingly more concerned with what contents are available than about the location of resources [Cis17]. Consequently, ICN treats data as fundamental by naming and addressing data objects at the network level, which represents a significant departure from the Internet's host-centric architecture. By decoupling content from location in this way, ICN naturally supports features like in-network caching and many-to-many communication. In ICN, a request from a user application carries the name or identification of the desired data object; the network then tries to answer the request with the data object(s) that best match it. The name can also refer to a host machine or a location, making ICN more general than the host-to-host communication model.

Although the many recent ICN projects differ in their design details, they share the same objec-

tive of developing an architecture for efficient information delivery by making content directly addressable [XVS+14, ADI+12]. For this research, we focus on the Named Data Networking architecture [NSF, ZAB+14], one of the prominent ICN approaches, for two primary reasons:

i NDN incorporates a stateful and intelligent forwarding plane in its architecture, referred to as the *forwarding strategy module*, for directing requests. This presents a useful framework for exploiting learning algorithms and adaptation in the network.

ii Secondly, the project handlers provide an open-source simulator for extensive experimentation of NDN-based networks.

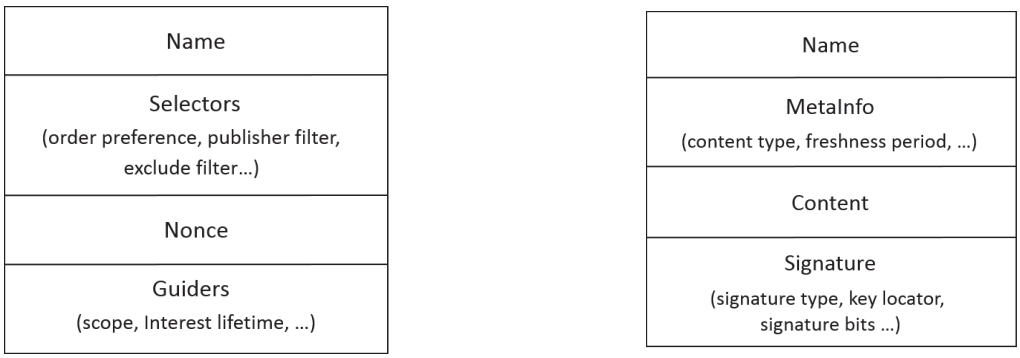
In the remainder of this chapter, we describe the NDN architecture and its key design decisions, including the simulation tool in Section 4.2. Section 4.3 briefly describes some of the other ICN initiatives. Finally, we outline the models and assumptions for the information-centric networks we evaluate in Section 4.4 and conclude the chapter in Section 4.5.

4.2 Named Data Networking

NDN developed from an earlier project, Content-Centric Networking (CCN) [JST+09], with which it shares similar basic ideas. The aim of the project is to develop an entirely new Internet architecture which benefits from the successes of the current one, in addition to addressing its weaknesses especially with regards to content delivery. NDN is also designed to be compatible with Internet Protocol (IP) to facilitate its deployment and evolution. The project was funded by the National Science Foundation (NSF), under its Future Internet Architecture Program. NDN has been successfully applied in real-time communication [GB15], sensor networks [Che14, SRS15], and vehicular communication [GPP+14]. A comprehensive survey of NDN applications is presented in [SRS+16].

4.2.1 Packets

NDN uses two types of packets for communication: *Interest* and *Data* packets. Requests are issued using Interest packets, and Data packets are used to carry the requested contents. Both packets include the name or identification of the desired data object as shown in Figure 4.1. An Interest is uniquely identified by its name and Nonce values, where the Nonce, which takes a non-negative integer value, is randomly generated by the requesting application. In addition, the Data packet also includes a digital signature, created by the content producer, that binds the name with the content.



(a) Interest (b) Data

Figure 4.1: NDN packet formats.

NDN currently adopts the Type-Length-Value (TLV) encoding scheme to encode its packets, so that an NDN packet is a collection of nested-TLVs. The outer most TLV specifies whether the packet is an Interest or a Data packet.

4.2.2 Names

NDN allows the name in a packet to refer to any piece of content, a command or an endpoint. It also allows each application to choose the naming scheme suitable to it, as long as the routers can recognise the boundaries in a name if it has multiple components. The NDN design currently adopts hierarchically structured names similar to URLs. For example, the name ***/icl/electrical-engineering/study/phd/*** can refer to a page which has information about the application process for a Ph.D. programme in the Department of Electrical and Electronic

Engineering at Imperial College. This particular example has 4 name components separated by the forward slash. As with IP addressing, the hierarchical structure allows for name aggregation which can improve the scalability of the routing system. It is important to mention that the namespace management is central to any content-centric architecture and is an active area of research in ICN [XVS+14, BCA+12].

4.2.3 Routing and Forwarding

An NDN content router (CR) maintains three data structures for name resolution and forwarding of Data: a Content Store (CS), a Pending Interest Table (PIT) and a Forwarding Information Base (FIB) in addition to the forwarding strategy module.

The CS acts as a temporary memory for Data. It is used to implement in-network caching and can satisfy matching requests. A CR caches visiting Data packets according to a well-defined caching policy. Due to the limited memory capacity, the caching policy will also include a replacement strategy for evicting contents from the CS to make room for new ones. Although the NDN project does not propose any caching policy, the default policy assumed caches a Data packet at every CR along its path.

The PIT is used to keep track of the unanswered Interests. Each entry in the PIT represents an Interest forwarded upstream towards content sources. This information is also useful in directing the Data packets. Furthermore, the PIT also remembers recently satisfied Interests for loop detection and performance measurement purposes. Hence, in more recent literature, the term *Interest Table* is used in the place of the *Pending Interest Table*.

The PIT also aggregates information about subsequent Interests that arrive and match any of its entries; the CR then decides whether or not to forward such Interests. Each PIT entry is identified by an Interest and contains a Nonce list, a list of arrival and outgoing links, and the most recent arrival and forwarding times on these links. Since Interests can be uniquely identified, the PIT can detect when an Interest loops. Moreover, by keeping the forwarding state in the PIT, delivery performance can be estimated and used by the strategy module in

influencing future forwarding decisions.

Finally, the FIB contains a mapping of the reachable content names to the outgoing interfaces of the CR. The FIB is populated and updated by a name-based routing protocol. Most of the intra-domain name-based routing protocols proposed for NDN [WHY+12, HAA+13, HGLA15] are inspired by the conventional Open Shortest Path First (OSPF) protocol used in IP networks. The strategy module is a program defined for each reachable name announced by the routing protocol, and it is expected to leverage on both the routing information and the observed content delivery measurements in forwarding the Interests.

Figure 4.2 shows the major data structures of the NDN forwarding engine model. In this illustra-

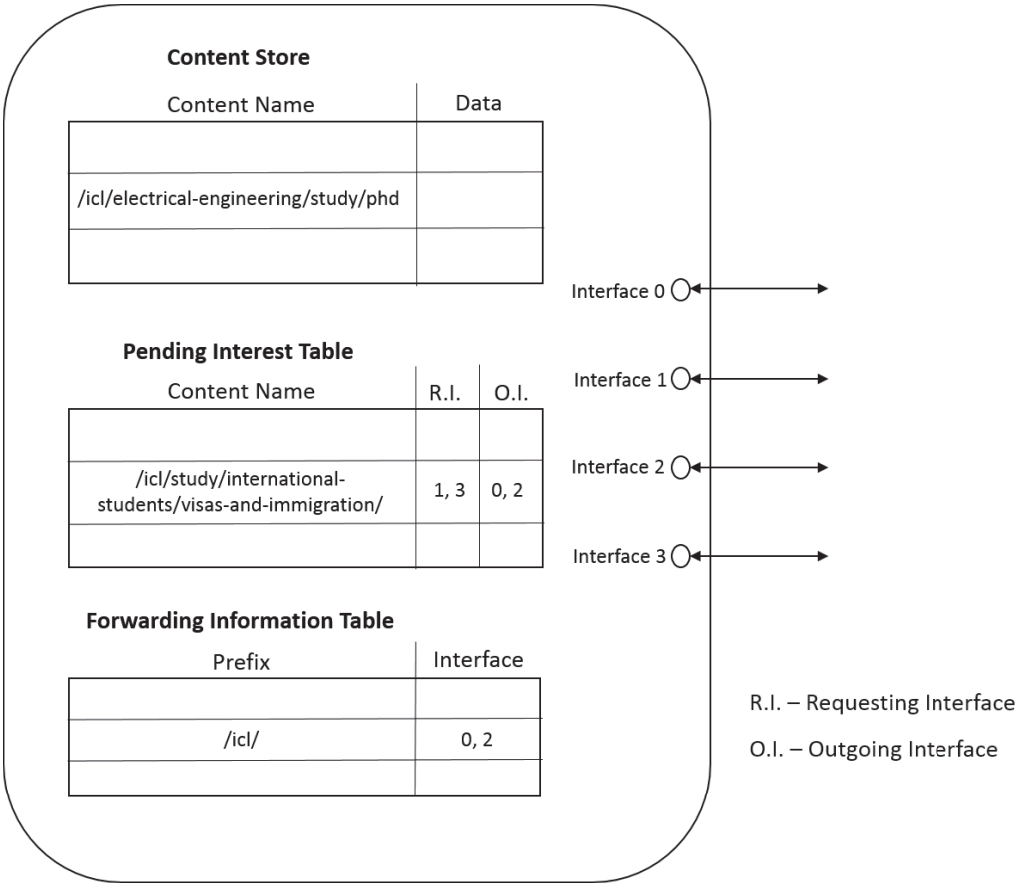


Figure 4.2: NDN Forwarding Engine Model.

tion, we assume that the machine has 4 interfaces and the page ***/icl/electrical-engineering/study/phd/*** currently cached in its CS. Consequently, the CR can immediately satisfy an Interest for the information about the application process for a Ph.D. programme in the Department of Electrical and Electronic Engineering at Imperial College. The PIT enty shown in Figure 4.2 indicates

that the CR is expecting data for the page ***/icl/study/international-students/visas-and-immigration/*** on the interfaces 0 and 2, following the arrivals of Interests on the other interfaces 1 and 3. Therefore, when the Data packet carrying the requested information arrives, it will be sent on the 2 requesting interfaces. Finally, Figure 4.2 shows a FIB entry which indicates that the CR knows 2 interfaces, interfaces 0 and 2, for forwarding Interests with content names prefixes by ***/icl/***.

When a CR receives an Interest, it first checks its CS for matching data objects. According to the blueprint for the NDN architecture in [JST+09], Data in the CS is said to match an Interest if the content name of the Interest is a prefix of the content name of the Data. NDN also allows the requesting application to have some control over which Data packets are satisfactory through an optional *selector* field in the Interest packet. In the event of a match, the Data packet is returned on the arrival interface of the Interest packet. Otherwise, a lookup, which should also consider the entries in the selector field, is performed in the PIT for an exact match. A match in the PIT means the CR is already expecting a response that will satisfy the received Interest, so the corresponding PIT entry is updated and the strategy decides whether or not to still forward the Interest packet. Otherwise, the longest-prefix match entry in the FIB is found, and the strategy module decides the outgoing interface(s) to forward the Interest. If the lookup in the FIB fails, the CR has no knowledge of the desired content and the Interest is discarded. Figure 4.3 summarises the Interest forwarding process at a CR.

Data packets, on the other hand, are forwarded along the reverse path used by their corresponding Interest packets by using the information recorded in the PITs. On receiving a Data packet, the CR performs a lookup in the PIT for a matching entry. If there is a match, the CR sends a copy of the Data packet on all the interfaces listed under “incoming” in the entry. The CR then decides whether or not to cache the Data packet. Otherwise, the content is considered unsolicited, and the Data packet could be discarded without caching. Figure 4.4 summarises the Data forwarding process at a CR.

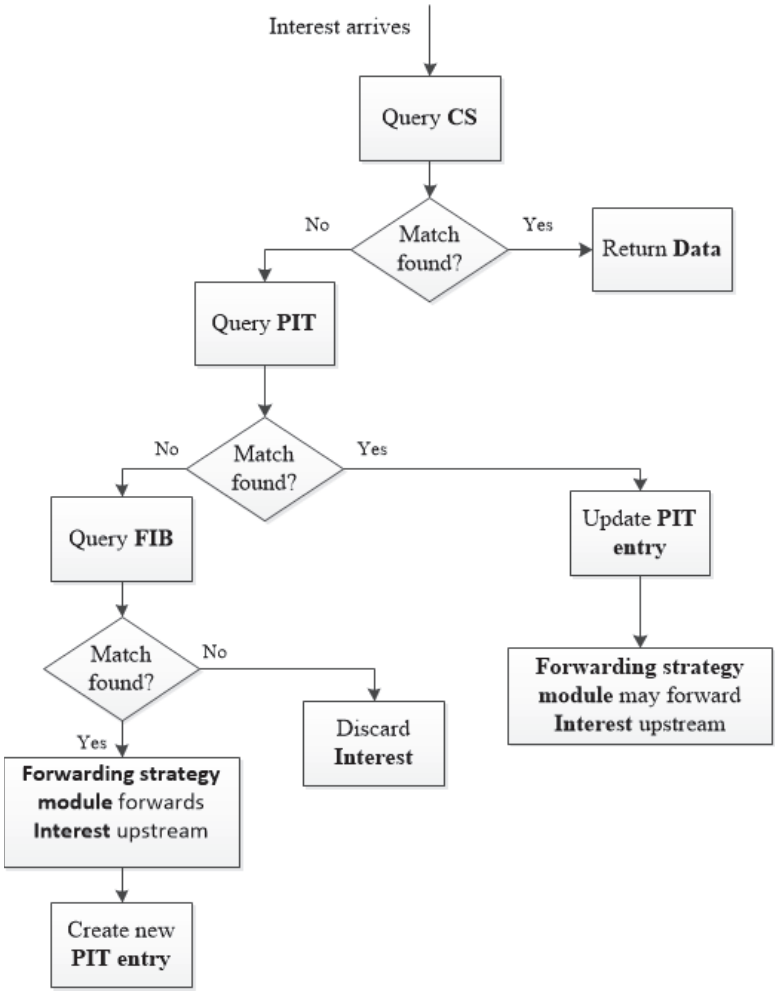


Figure 4.3: Interest forwarding process at an NDN content router.

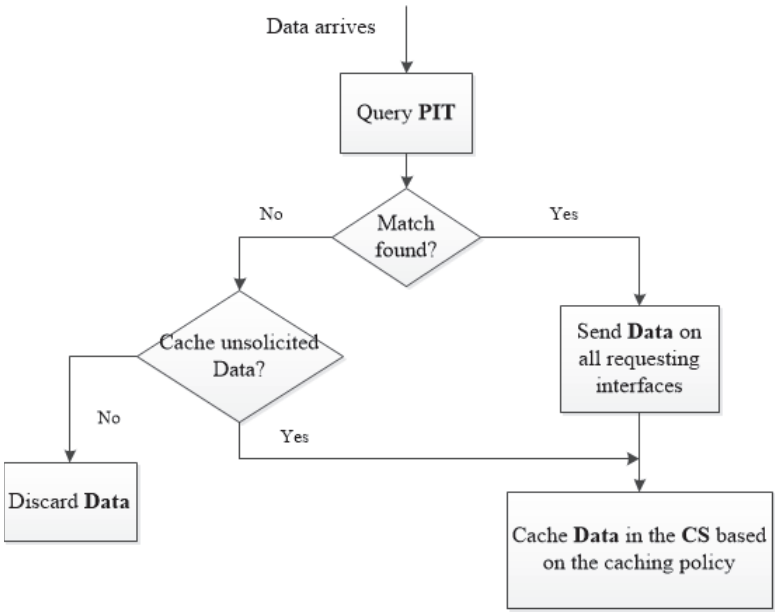


Figure 4.4: Data forwarding process at an NDN content router.

4.2.4 Reliability and Flow Control

In general, communication in ICN is driven by the receiver. As a result, the requesting application is ultimately responsible for reliable delivery and flow control.

With respect to reliability, the requesting application should monitor unanswered Interests and retransmit if the content is still desired. Retransmissions are also possible at the CRs since the PIT keeps flow state. To avoid wasting network resources, the strategy module can control retransmissions by restricting them to within a given period after the arrival of the initial Interest. *Interest lifetimes* were introduced in [YAW+12, YAM+13] so that an application can specify how long an Interest can remain pending at a CR. Once the Interest lifetime expires, the PIT entry is removed.

NDN also introduced Interest NACKs to address the inefficiencies that result from the dangling states in the PIT caused by unsatisfied Interests [YAW+12, YAM+13]. When it cannot forward nor satisfy an Interest, a CR responds with an Interest NACK; the Interest NACK also carries a code describing the reason(s) forwarding was impossible. Therefore, Interest NACKs can help the network to quickly and informedly detect faults and, when possible, try other forwarding options.

Without packet losses, a one-to-one flow balance is maintained between an Interest and a response packet. In other words, an Interest retrieves at most either a Data packet or an Interest NACK. As a result, a congestion control scheme can be implemented hop-by-hop [YAW+12, YAM+13, SYZZ16]. An upstream node that becomes congested can inform downstream nodes including the requesting application using the Data packets or Interest NACKs. Downstream nodes would then respond by reducing their rates of sending Interests upstream or by diverting subsequent Interests to alternative paths or a combination of both actions.

4.2.5 Security

As one would expect, ICN adopts a data-centric approach to security. Rather than securing the end-points and the connections as in conventional networking, it is the data itself that is secured. As a result, CRs and consumers can validate a piece of content irrespective of how or where it was obtained. Embodying security in content is generally implemented in ICN via the naming structure.

In NDN, publishers are required to cryptographically sign every Data packet. The signature binds the content name and its corresponding content. NDN allows a publisher to choose the signature algorithm that best suits its needs. A pointer for retrieving the public key used to sign the packet is also provided in the Data packet. Network nodes can then verify each content by checking that the name-content mapping was signed by the retrieved key. The report in [ZYZ+18] addresses the challenges of this approach with respect to trust management. This is necessary so that in addition to checking the packet signature, consumers can confirm that the public key owner is a certified publisher. NDN also allows the encryption of content to control access to information.

The likely network attacks in NDN and how to mitigate against them have also been studied. The Interest flooding denial of service attack is probably the easiest to implement where an attacker rapidly generates and sends Interests to overwhelm the CRs. This can saturate the PITs and deny resources to other legitimate consumers. Another possible attack is the cache pollution attack. Here, an attacker with access to compromised routers in a network and with knowledge of the content popularity, globally or locally, continuously sends Interests following a 'malicious' pattern to degrade the delivery performance experienced by the honest users by reducing the local cache hit possibilities. For instance, the attacker could generate Interests for the least-popular contents so that they can dominate the cache space. Collaborative methods to counter flooding attacks are proposed in [JST+09, CCGT13, AMM+13] while [CGT13] addresses the cache pollution attack.

4.2.6 Mobility

The content-centric communication principle and the receiver-driven style of communication enable the NDN architecture to intrinsically support consumer mobility. A mobile consumer that changes its location after issuing an Interest can simply resend it as soon as it regains network access at its new location. The only cost to the network will be the resources expended in retrieving the content to the old location.

Publisher mobility, on the other hand, is more complicated to support because the routing tables might need to be updated to reflect the current location of the publisher. A popular approach that has been studied to solve the publisher mobility problem adapts ideas from Mobile IP [PJA11]. In this approach [WWK13, ZZZ14, CZM+17], static registration servers are used to track the movement of publishers and to redirect traffic to their current points of attachment.

Rather than track the publishers, in [PAR+18], the servers act as data repositories or depots where mobile publishers offload their contents. The data repositories are then responsible for satisfying the requests meant for the mobile publishers.

4.2.7 NDN Simulator

As part of the NDN project, the handlers have also developed an open-source simulator, called *ndnSim* [MAZ17], for experimentation. *ndnSim* provides a common high-fidelity platform for evaluating the NDN architecture. The simulator package is based on the NS-3 framework [NS-], which is an open-source discrete-event network simulator written in C++ and designed primarily for research purposes. *ndnSim* realizes the full NDN network stack, that is, the NDN Forwarder Daemon (NFD) [ASZ+18] and supporting libraries. The NFD includes implementations of the data structures for name resolution and Data forwarding: CS, PIT, and FIB. It also implements some of the conventional caching policies and replacement strategies, and some of the proposed strategy modules from the literature. Moreover, *ndnSim* continues to undergo design changes and development according to the advancements of the NDN research.

For our purposes, we have made additions to the NFD to evaluate our proposed solutions. We have used the ndnSim version 2.5 which is a recent version considering that the version 2.6 was released at the time of writing the report. We installed ndnSim on 3 Linux workstations: two with Intel Core i5-2500 CPU @ 3.30GHz having 12 GB DDR3 RAM and one with Intel Core i7-3770 CPU @ 3.40GHz having 8 GB DDR3 RAM.

4.3 Other ICN approaches

Just like NDN, several other projects exist that focus on the design of ICN architectures as alternatives to the host-centric communication model. In this section, we briefly describe 2 other prominent ICN projects, focusing on their approaches to naming, and routing and forwarding of requests and data.

4.3.1 Data-Oriented Network Architecture (DONA)

DONA [KCC+07], from UC Berkeley, was proposed as an overlay architecture to address naming and name resolution on the Internet, so it retains IP addressing and routing.

DONA uses two types of packets: *FIND* packets sent by consumers to locate named objects, and *REGISTER* packets sent by publishers to announce the availability of a named object. Furthermore, a publisher assigns to each published data object, flat, self-certifying and globally unique names. DONA names have 2 components: a *principal* and a *label*. The principal is the cryptographic hash of the publisher's public key, while the label uniquely identifies an object with respect to the publisher.

The architecture comprises of specialised servers, called *Resolution Handlers* (RHs), such that there is at least one RH at each of the Autonomous Systems (AS) on the Internet. The RHs form an overlay responsible for resolving name requests. A publisher announces the availability of an object by sending a *REGISTER* packet with the object's name to its local RH. Leveraging on the relationships of interconnected domains on the Internet, an RH forwards a *REGISTER*

packet to its parent and peering domains. This guarantees that all *REGISTER* packets reach all the RHs in the tier-1 ASes. On receiving a *REGISTER*, each RH also records a mapping between the object's name and the sender of the packet, which could be another RH or the publisher. Therefore, an RH forwards each *FIND* packet towards either tier-1 RHs or the RH closest to the publisher of the requested object. The requested object is sent either directly to the IP address of the customer using the regular IP routing or along the same path used by the corresponding FIND packet.

4.3.2 Publish Subscribe Internet Technology (PURSUIT)

In PURSUIT [FP7c], information items are organised into scopes identified by *scope IDs* (SId). Scopes are network domains organised hierarchically, and each scope defines the domain over which the information items assigned to it are available. As a result, an information item is identified by a SId, and a unique identifier, the *rendezvous ID* (RId), within the given scope. SIdS and RIds use flat names as in DONA.

The PURSUIT architecture comprises of 3 types of nodes for implementing the routing and forwarding functions: *Rendezvous Nodes* (RNs), *Topology Manager* (TM) nodes and *Forwarding Nodes* (FNs). RNs are responsible for tracking the published information items, and SIds are mapped, one-to-one, to the RNs via a hierarchically distributed hash table (DHT). TM nodes are used to create connections between a consumer and a publisher. Finally, FNs are the regular store-and-forward routers.

PURSUIT uses 3 types of packets: a *PUBLISH* packet, a *SUBSCRIBE packet*, and a *START PUBLISH* packet. A publisher advertises an item by sending a *PUBLISH* packet with the item's name to its nearest RN. *PUBLISH* packets are always routed by the DHT to the RN assigned to the SId in the packet. A consumer issues a SUBSCRIBE packet to its nearest RN to locate an information item. Just like *PUBLISH* packets, a *SUBSCRIBE* packet is sent to the RN assigned to the SId in the packet. On the arrival of a *SUBSCRIBE* packet, the destination RN will then inform its nearest TM node to set up a connection between the consumer and the

publisher of the requested item. Finally, the full path information computed by the TM is sent to the publisher using a *START PUBLISH* packet and communication begins.

4.4 System and Scenario Description

To conclude this chapter, we use this section to give an overview of the models and assumptions we adopt in our experimentations with ICN-based networks. While there is no complete consensus on the evaluation of information-centric networks, the scenarios we consider in our simulations are in line with much of the related literature, especially with respect to the generated workload and the network topologies used.

4.4.1 Request arrival model

The most common approach in analysing caching systems is the assumption of the independent reference model (IRM) [CD73] to model the arrival of requests, with a Zipf-like distribution to represent the content popularity. The IRM describes the requests arriving at a cache for a fixed catalog of items as forming a sequence of independent, identically distributed random variables. In other words, the probability that a request is for a given item is a constant and independent of the past requests.

The IRM has the advantage that it is simple and has enabled the development of tractable models in the analysis of caching systems [CTW02, DT90, Gel73a]. Although IRM does not adequately capture phenomena like *temporal locality* and *spatial locality* which have been shown to exist in real traces [TAG+13, ABCdO96], other studies [BCF+99, JB00] argue that their long-term effect can be well accounted for by the careful choice of a Zipf-like distribution to represent content popularity. Temporal locality occurs when recently accessed items are more likely to be requested in the near future. Spatial locality, on the other hand, describes the situation where the request for an item becomes more likely because a similar item was referenced in the recent past.

Our adoption of the IRM to model the arrival of requests is primarily because of its popularity in the literature. However, another technique that can be used and which we plan to explore in future works is based on Belady's Life-Time Function [Bel66a, Gel73b] which characterises the probability of not finding a data object as a function of the size of the total cache space size. Some of these techniques are also detailed in [GM10].

4.4.2 Content popularity and catalogue size

The choice of the popularity distribution is an important factor that determines the performance of in-network caching in ICN. The Zipf's law is the favoured choice to model popularity in ICN [MCG11, CGMP11, RR14, BSF15].

Zipf's law, popularised initially in Statistical Linguistics, predicts, for a catalog of N items, that the probability of referencing the i-th most popular item is

$$\rho_N(i) = \frac{\frac{1}{i^\alpha}}{\sum_{i=1}^{N} \frac{1}{i^\alpha}}, \tag{4.1}$$

where the *skew factor*, α, is a constant characterising the distribution. The request arrival pattern is significantly influenced by the skew factor as it determines the size of the subset of a catalogue of contents that receives a given proportion of the requests. We illustrate the influence of the skew factor in Figure 4.5 using the plots of the cummulative probability distribution of requests against the top $r\%$ of the contents in popularity, considering a catalogue of 10^6 contents. Figure 4.5 compares the cummulative distribution for 4 different values of the skew factor, $\alpha = 0.6$, 0.8, 1.0 and 1.2. The figure shows that the top 10% most popular contents account for about 40%, 61%, 84% and 96% of all requests for $\alpha = 0.6$, 0.8, 1.0 and 1.2, respectively. The implication for a caching system is that as the skew factor of the Zipf distribution increases, the subset of the catalog that receives a significant proportion of the requests becomes smaller, and, as a result, the more effective caching will be. In fact, for $\alpha = 2.4$, a cache that can store 0.01% of the whole catalog (which is, 100 contents) can satisfy more than 99.92% of all the requests if it caches the most popular contents.

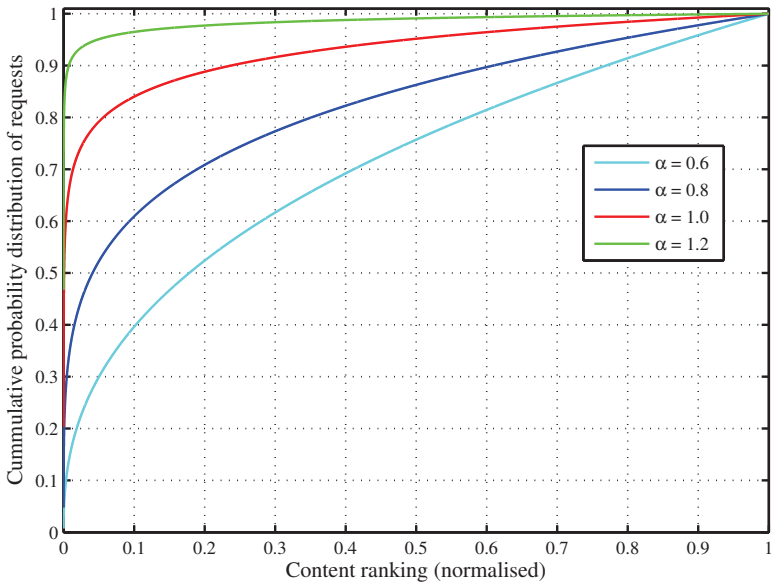


Figure 4.5: Cummulative probability distribution of requests to the top $r\%$ most popular contents. Skew factor values of $\alpha = 0.6$, 0.8, 1.0 and 1.2 and a catologue size $N = 10^6$ are considered.

The skew factor values used in the literature are informed by those arrived at from large-scale studies using the actual traces observed at ISPs and CDNs [BCF+99, FLT+13, CKR+07, HS08]. The authors in [BCF+99] and [FLT+13] observed α values in the range $[0.6 - 1.04]$ from their studies of 6 web proxy traces and CDN request logs in 3 geographical locations, respectively. A similar analysis of Youtube content popularity in [CKR+07] showed good fittings for α values of 2.3 and 2.5 for two representative categories of Youtube videos.

On the other hand, some other works have adopted the more general Mandelbrot-Zipf (Mzipf) distribition in their analysis of information-centric networks [RR11, KXP11]. In MZipf,

$$\rho_N(i) = \frac{\frac{1}{(i+q)^\alpha}}{\sum_{i=1}^{N} \frac{1}{(i+q)^\alpha}}, \tag{4.2}$$

where α and q both characterise the distribution. Equation (4.2) reduces to (4.1) when $q = 0$. This is supported by the analysis in [HS08] of P2P traffic in different ASes which showed the MZipf distribution as a better fit than the Zipf-like distribution. Values of $\alpha \in [0.6, 0.78]$ and $q \in [3, 121]$ were obtained from their analysis. In Figure 4.6, we try to show the impact of the q parameter of the MZipf distribution to the request pattern by considering the proportion of the requests that go to the top 10% most popular contents in a catalogue of 10^6 objects. The figure

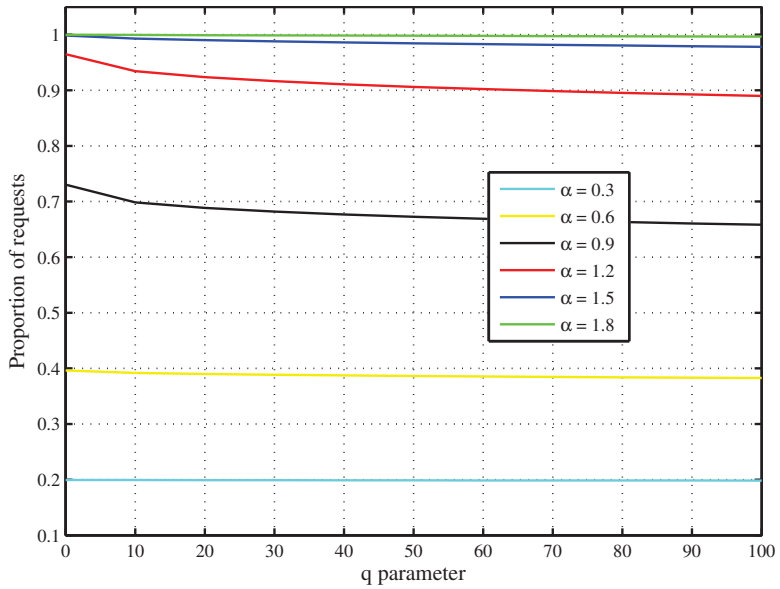


Figure 4.6: Effect of the q parameter of the MZipf distribution on the request pattern. The plots show the proportion of the requests accounted for by the top 10% most popular contents in a catalogue of 10^6 objects for skew factor values of: $\alpha = 0.3, 0.6, 0.9, 1.2, 1.5$ and 1.8.

shows that, in general, for each of the considered α values, increasing the q parameter reduces the proportion of requests sent for the top 10% most popular contents. But, the influence of q on this proportion reduces as q increases. Furthermore, the influence of q is almost negligible for α values outside the range $[0.6 - 1.5]$.

In our tests, we adopt the MZipf distribution to represent content popularity. However, most of the evaluations using the *ndnSim* do not scale well with the catalogue size. That is, the larger the catalogue size, the more computationally expensive a simulation run will be. As a result, we have scaled down these tests in terms of the catalogue sizes considered. Since the studies of content popularity are usually conducted with catalogue sizes of the order of 10^8, it is relevant to examine the impact of N on the request pattern of the MZipf distribution. Figure 4.7 compares the request patterns for four different catalogue sizes ($N \in 10^3, 10^4, 10^5, 10^6$) using the size of a given subset of the most popular content as the reference. The vertical axis of Figure 4.7 represents, using a log scale, the number of contents in the subset of a catalogue comprising of the most popular contents that account for 80% of the requests. The size of this subset is measured for different MZipf distributions with skew factor values $\alpha \in 0, 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.4$ and q fixed at 5.0. As expected, the figure shows that scaling down the catalogue size significantly affects the request pattern for smaller values of α ($\alpha < 1$).

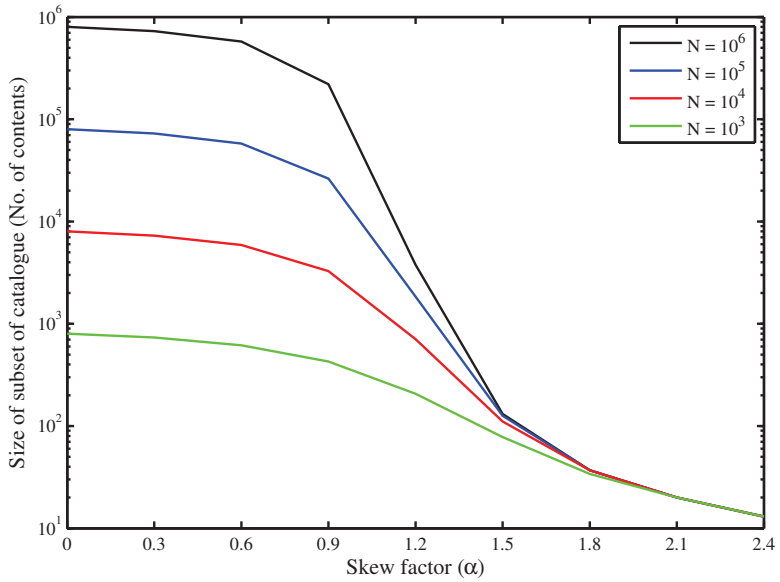


Figure 4.7: Effect of the catalogue size (N) of the MZipf distribution on the request pattern. The plot shows the number of contents in a given subset of the catalogue which comprises of the most popular contents accounting for 80% of all the requests for different MZipf distributions. The following parameter settings are used for the plots: $N \in \{10^3, 10^4, 10^5, 10^6\}$, $\alpha \in \{0, 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.4\}$ and $q = 5.0$.

However, the curves converge to the same value at around $\alpha = 1.8$.

Therefore, to improve the fidelity of our tests under the computational restraints, we consider skew factor values in the range [0, 1.8] and small memory sizes at the caches for reasonable cache size to catalogue size ratios. Uniform popularity ($\alpha = 0$) has also been considered in some other works [SGFT13, RK09, PRH17].

4.4.3 Network topology

With respect to the network segments used in our tests, we have also followed the trend in recent ICN studies by adopting real ISP topologies. Furthermore, we assume that the nodes and links within a topology have identical resources in terms of cache allocation and link characteristics. By identical link characteristics, we mean that the links in the topologies we consider have the same bandwidth and delay.

Finally, we assume in all the tests carried out that the system operates below congestion. This assumption informs the choices of the link characteristics and the rate of arrival of requests

into the network. While some of the measures we propose in this work can deal with network congestion, we have not considered a congested network because communication in NDN is receiver-driven, so that adjusting the rate of sending requests at the receiver is the best solution for congestion. Therefore, our congestion assumption allows for simplified tests for an initial evaluation of our proposals. Future work will focus on the effects of congestion on our proposed system.

Table 4.1 describes the network topologies used for the ICN simulations and Figure 4.8 shows the Elibackbone topology according to the dataset provided in [KNF+11].

Table 4.1: Network topologies for ICN simulations.

	Number of nodes	**Number of links**
Abilene	11	14
Elibackbone	20	30
Geant	40	61

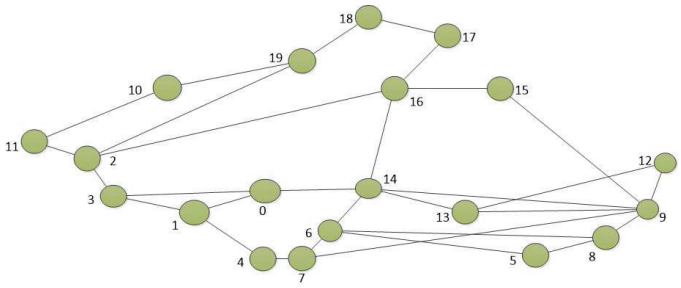


Figure 4.8: *Elibackbone* topology. The network consists of 20 nodes and 30 links.

4.5 Conclusion

In this chapter, we presented the NDN architecture which we have adopted for our ICN analysis. We summarised the main design decisions of the NDN project including the naming of objects, routing and forwarding, transport, security, and mobility. Furthermore, we described the NDN simulator, *ndnSim* which we have used to evaluate NDN-based networks. Finally, we outlined the models and assumptions for the information-centric networks we evaluate.

Chapter 5

On-demand Adaptive Cache Management in Information-Centric Networking

5.1 Introduction

Decoupling content from location is fundamental to the ICN concept. This is similar to applying to the "names" of ICN, the "virtual memory" concept that is commonly used in operating systems [Den97, BL17]. ICN achieves this separation by naming and addressing data units at the network level, so that request packets are routed and satisfied based on the name or identification of the desired content unit, rather than using an address. As a result, an ICN-based network can easily support in-network caching and replication. That is, the network forwarding entities are able to cache data objects and satisfy the requests regarding the contents in their cache, which is vital to many of the proposed ICN approaches [XVS+14, ZLZ15].

In-network caching presents fresh challenges and opportunities to network administrators, since it is in their interest, both for customer experience and transit costs, to serve as many requests as possible locally. Also, administrators could offer publishers special services whereby their contents receive priority in the allocation of the cache space. Developing such a dynamic system for caching decisions in a network of caches is what we address in this chapter.

The efficient use of limited caching space is not new and has been well studied in computer memory management [Gel73a, ADU71, Bel66b, Gel73b], web caching [PB03] and content distribution networks (CDNs) [BGW10, KRR02]. In the ICN context, there is also an extensive body of research focused on developing policies for deciding the right cache allocations [RR12, WLT+13, DBGLA14, FLT+13], the favourable contents to cache and where to place them, and the selecion of data to replace in a network of caches [CHPP13, JST+09, RR11, GXS12, SGFT13, IY17]. Proposed caching strategies range from simple algorithms where routers make caching decisions independently [JST+09, RR11], to cache management systems, requiring high levels of coordination, which implement near-optimal solutions for content placement in a network [GXS12, SGFT13, IY17]. These systems combine the knowledge about the content request patterns and the network topology in deciding the best contents to cache and the best place to cache them so as to optimise a given cost function.

Most of the proposed cache management systems for ICN have approached this problem as an offline cache planning problem like in CDNs. Their solutions are either static or result in the pushing of contents into the caches periodically following an observed request pattern [GXS12, SGFT13, IY17]. In contrast, we propose a cache management system able to deploy an online adaptive caching strategy in an ICN-based network. Our proposed system works on-demand such that caching decisions follow the user requests. This approach enables caching decisions to be adapted in real-time and in a pragmatic way through the controlled reconfiguration of the network.

In this chapter, we analyse the content placement problem as an "on-demand" problem and then present approximate solutions for optimising a given network-wide cost. Furthermore, to implement our solutions, we propose a framework which comprises coordinating cache managers at the access points of a network and with a global view of it. By virtue of their location, these managers can detect cache misses when they occur in the network and be prompted to make caching decisions that optimise a given objective.

The remainder of this chapter is organised as follows. A review of the relevant literature is presented in Section 5.2. In section 5.3, we outline the cache optimisation problem as an on-

demand problem and present possible heuristics for its solution. Section 5.4 outlines a cache management framework for implementing the proposed optimisation algorithm, specifically on an NDN-based network. Simulations results and discussions are presented in Section 5.5, and we conclude the chapter in Section 5.6.

5.2 Related Work

Caching strategies can be classified as either on-path or off-path strategies. In on-path caching, contents are cached at the nodes along the paths towards their publishers. We refer to such paths as "routing paths" since most ICN architectures construct them in order to forward requests. However, in off-path caching, contents can be cached outside the routing paths. Another important classification of caching strategies in ICN is cooperative versus non-cooperative strategies. Routers coordinate in making caching decisions in the cooperative strategies by sharing relevant information which could include their entire caching state. But, in non-cooperative strategies, each router makes caching decisions independently.

In the ICN architectures where a request packet and its corresponding data packet use the same path, on-path caching is easily implemented by allowing the caching decisions to be made as the data packet travels towards a user. The caching decisions in on-path caching can either be deterministic or probabilistic. On-path caching strategies generally require little or no coordination among the routers, and because caching is along the routing paths, no additional signalling is necessary to update the forwarding tables.

The default caching strategy in the *Named Data Networking* (NDN) architecture, one of the prominent ICN projects, is the Leave Copy Everywhere (LCE) policy [JST+09]. In LCE, a data packet is cached at every router it visits resulting in high rate of eviction, which can cause cache pollution [RR11]. The Leave Copy Down (LCD) policy, where a content is only cached a hop downstream from the router it originates from, is tested in [RR11, RR14]. In [SPFT09], a data object is cached only at the closest node to the requesting application. Cho et al. propose *WAVE* [CLP+12] which is similar to the LCD; however, the upstream router

suggests the chunks of a file that the downstream routers can cache. Their aim is to quickly move the chunks of popular contents to caches closer to the users as the access rates for the contents increase. The caching policy proposed in [CHPP13] is based on the betweenness centrality concept, which is a measure of centrality in graph theory based on shortest paths. In this scheme, only the routers with the highest betweenness centrality values along the delivery path cache the content. A request packet records the highest betweenness centrality value as it traverses the network so that its corresponding data packet is then cached at the routers with matching values. Rossi and Rossini, in [RR12], went further by using several graph-related centrality metrics in allocating a given cache amount among the nodes in a network.

On the other hand, in the probabilistic caching policies, routers cache contents according to some set probability. The LCE policy can be considered as a special case of probabilistic caching where the caching probability is always one. While arbitrary probabilities, smaller than one, were evaluated in [CHPP13, ANO10] and shown to be more cache efficient than the LCE, other works have proposed computing the caching probability while taking into consideration factors like the network topology and content popularity. Psaras et al. proposed the *ProbCache* scheme in [PCP12], where the caching probability when a content arrives at a node is computed by weighing the total caching capacity along the remainder of the delivery path with the relative position of the node along this path. As a result, the caching probability increases as the content moves towards the user. *PopCache* [STJ+13] weighs the router's relative position along the delivery path with the router's probability of requesting for the most popular contents up to its caching capacity.

Besides simplicity, another important benefit of on-path caching strategies is that they offer the same guarantees for content retrieval as the name-based routing protocol that contructs the delivery paths, and they can avoid unnecessary re-routing of request packets when the delivery paths are loop-free. However, caching here is opportunistic, and inefficiencies from high cache redundancies as a result of little coordination among the nodes can limit their delivery performance.

Off-path caching strategies attempt to address the deficiencies of on-path caching. They ex-

plore cooperation among routers to reduce redundancy and improve the efficient use of the caching space. By caching outside the routing paths, these strategies usually require additional signalling to coordinate caching and forwarding.

In addition to an on-path caching strategy, in [RPS16], routers also forward the evicted data objects and those that cannot be cached to a central router for off-path caching. The authors assumed an overlay ICN which includes ICN managers that resolve network paths and are updated with the caching and replacement operations in the network.

In [WGMK11], routers record data packet movements, that is, the incoming and outgoing interfaces, in a table which is then consulted in diverting subsequent requests towards the neighbouring caches with a good probability of having the content. The authors improved their neighbourhood search algorithm in [WWK12] by using bloom filters to exchange cache state information among neighbouring routers. Also, to reduce cache redundancy, a data packet is marked once it is cached so that no other router along the path caches it; although, the authors do not make it clear which router makes the caching decision..

In [LS11], each router in a network is assigned a unique positive integer label, and via modulo operation, chunks of contents are mapped one-to-one to each router such that a router can only cache the chunks that match its label.

The reality is that the competition inherent at the core of the Internet means that the level of cooperation required for off-path caching is more attainable among a network of nodes under the same administrative control. As a result, cache management frameworks for realising adaptive and near-optimal placement and retrieval of contents within a network have been studied. The optimal content placement or replication problem is generally posed as a combinatorial optimisation problem of maximizing a linear cost that is weighted by the observed or predicted request pattern and the costs of the available paths, subject to capacity constraints. Ref. [KRR02] showed the problem to be NP-complete, so heuristics have been used to obtain approximate solutions [KRR02, QPV01].

An optimal distribution of popular contents among the routers in a network, where the router

with the lowest average access cost from within the network, caches the most popular contents until the caching space is fully assigned, is adopted in [GXS12]. The routers coordinate to rank both the most popular contents the network is able to cache, and the network routers based on their access costs. With this information and the assumed distribution of contents, routers have full knowledge of the contents that can be delivered from their other cooperating routers, and requests can be forwarded accordingly.

In [IY17], a joint optimisation of routing and caching in ICN is presented as a submodular problem under matroid constraints, and a distributed adaptive algorithm which is expected to converge to within a constant approximation from the optimal solution.

An additional architecture for deploying network-wide cache policies according to real-time observed request patterns is proposed in [SGFT13]. The architecture comprises of distributed cache managers which are mapped one-to-one with the cache-enabled routers in the network. This allows them to coordinate and exchange relevant network information to optimise a given objective.

Software-defined networking (SDN) has also been put forward as a possible solution for implementing cache management in ICN [WRL+14, CW13]. Through the SDN architecture, a controller with a global view of the network can deploy an optimal caching strategy and nearest replica routing.

Rather than optimising cache placement, the approaches in [SLYJ15, BKST13] focus on increasing content availability within the network with minimum signalling overhead. To achieve this, contents are mapped to each router in a network by applying a hash function on the content identification, and only the designated router is allowed to cache a content. Forwarding of requests and data are then coordinated such that they traverse the designated routers while crossing the network.

5.3 Content Placement Problem

Consider a connected network $G = (V, E)$ in which a node $v \in V$ can either be an NDN router or a normal store-and-forward router. We denote by $N \subset V$ the subset of network nodes that are NDN routers or content routers (CRs). We have assumed that every NDN router has the capability to cache data units or contents that we call "pages" here for simplicity, and D is the set of all cacheable contents or pages.

We represent by $t = 0,\ 1,\ 2,\ ...$ the time, in discrete increments, when some user makes a request for some data page. $M_t(j)$ is the set of content pages that are stored in NDN router $j \in N$ at (just before) instant t. We assume that the set of cacheable pages is larger than the amount of storage available at all of the NDN nodes, i.e.

$$|D| > \sum_{j \in N} |M_t(j)|. \tag{5.1}$$

At an instant t, only one end user can request a page, and the variable $r_t \in \{V \times D\}$ denotes the request by some node for a given page, so that

$$\begin{aligned} r_t = (i, d)\ if\ at\ t\ an\ end\ user\ connected\ to\ i \in V \\ requests\ access\ to\ d \in D. \end{aligned} \tag{5.2}$$

We can also represent the requests probabilistically, so that we may indicate a relative frequency for a page d to be requested by a user connected to router $i \in V$:

$$\pi_t(i, d) = Prob[r_t = (i, d)]. \tag{5.3}$$

Also, define the binary variable $R_t(j, d) = 1$ if $d \in M_t(j)$ and $R_t(j, d) = 0$ otherwise, which represents whether page d resides in node $j \in N$. Similarly, we define $F_t(d) = \prod_{j \in N}[1 - R_t(j, d)]$ which takes the value 1 if d is in *none* of the NDN routers, and $F_t(d) = 0$ if d is in at least one of the NDN routers.

The system as a whole operates "on demand". In other words, we assume that a data item will not be fetched into any node unless it is requested; it is only loaded into a node in N when it is requested, and then it can be accessed by the user that requested it. Note that if some user that is connected to an NDN router i makes the request for d, the requested data item is not necessarily loaded into the memory of i, and the decision regarding the location will be taken by an optimiser that will consider the overall cost of accessing the data item.

The "on demand" transfer of d at time t occurs if just before t we have $F_t(d) = 1$. The decision denoted by $\delta_t(i, j, d) = 1$, indicates that the system decides to load d into the memory of the NDN router j when d is not already in some NDN router, and is requested by a user that is connected to router i. Note that all other $\delta_t(i, k, d) = 0$ for $k \in N, k \neq j$, so that $\sum_{j \in N} \delta_t(i, j, d) = 1$ and $\prod_{j \in N} \delta_t(i, j, d) = 0$, for all $t,\ i \in V,\ d \in D$.

Because of the limited size of the total memory that is available (5.1), during continuous operation when at each time unit some data item is requested, loading a new data page into any NDN node will mean that some other page must be removed from the same node.

Let $l(M_t(j)) \in M_t(j)$ be some designated element of $M_t(j)$, such as the "Least Recently Used", or the "First-In", or a randomly chosen element, which is removed for lack of space when a data item d is loaded into $M_t(j)$. Then for any $j \in N$ we have the following expression regarding the manner in which $M_{t+1}(j)$ depends on $M_t(j)$ and other quantities:

$$\begin{aligned} M_{t+1}(j) = \sum_{i \in V;\ d \in D} & 1[r_t = (i, d)]\{M_t(j)[1 - F_t(d)] \\ & + F_t(d)[M_t(j)(1 - \delta_t(i, j, d)) + \delta_t(i, j, d) \\ & (M_t(j) \cup \{d\} - l(M_t(j)))]\}, \end{aligned} \tag{5.4}$$

Now let $B(d, j)$ denote the cost, for instance in terms of transfer time, of bringing d from the repository to an NDN router j in the network. Also, let $T(j, i)$ be the cost of transfering some data item d from NDN router j to the end user at router i. Then the cost of satisfying the

request of a user at node i for a data item d at time t can be expressed simply as:

$$K_t(i,d) = \sum_{j \in N} \{ \ T(j,i)R_t(j,d) \ + \ F_t(d)\delta_t(i,j,d) \\ [B(d,j) + T(j,i)] \ \}, \tag{5.5}$$

since we will have a node-to-node transfer cost in the form of $T(j,i)$ when $M_j(t)$ contains d already, and if the page d was not in any of the NDN routers then it also had to be brought in from the repository and then transferred to i.

The "overall cost" of an access by the user at i to d will then be denoted by:

$$K_t(i,d) = \sum_{j \in N} 1[r_t = (i,d)]\{R_t(j,d)T(j,i) \ + \ F_t(d) \\ \delta_t(i,j,d)[B(d,j) + T(j,i)]\}. \tag{5.6}$$

Define $\pi_t(i,d) = Prob[r_t = (i,d)]$. Then the "average" or expected cost will be:

$$E[K_t(i,d)] = \sum_{j \in N} \pi_t(i,d)\{Prob[d \in M_t(j)]T(j,i) + \\ f_t(d)\delta_t(i,j,d)[B(d,j) + T(j,i)]\}, \tag{5.7}$$

where $f_t(d) = Prob[F_t(d) = 1]$.

Remark 1 A "local" or "greedy algorithm based" decision to optimise the system might choose to place the incoming data page d requested by node i in the NDN node j that has the smallest value of the transfer cost $T(j,i)$. Although this would have the advantage of placing data pages which are needed by i into the NDN node j that has the shortest access time from node i, this approach may result in frequent page faults (i.e. requests for pages that are not located in j) because each time a page is loaded into j some other page must be removed.

Remark 2 However, things are more complex because the placement choice will affect the "page fault probability" $f_t(d) = Prob[F_t(d) = 1]$ and the "page request probability" regarding different $j \in N$, which in turn can affect the "node request probability". Indeed, define the

node request probability as follows:

$$\begin{aligned}\rho_t(i,j) &= \sum_{d\in D} \pi_t(i,d).Prob[d \in M_t(j)], \\ &\quad where\ Prob[d \in M_t(j)]\ obviously\ depends\ on \\ &\quad \delta_\tau(k,j,d)\ for\ \tau < t,\ \forall\ k \in V.\end{aligned} \tag{5.8}$$

Thus, the optimal choice of $\delta_t(i,j,d)$ requires careful consideration.

5.3.1 On-demand Cache Management Algorithms

A better understanding of the optimisation problem can be gained by using a plausible model of the statistics of successive page requests. Thus in this section we assume that the variables $r_t \in \{V \times D\},\ t = 0,\ 1,\ 2,\ ...$ are independent and identically distributed random variables for successive values of t, and we can write $\pi_t(i,d) = p(i,d) \geq 0,\ \sum_{i\in V, d\in D} p(i,d) = 1$ since $\pi_t(i,d)$ does not depend on t when we assume that successive page request references are identically distributed. Also, we assume that $p_d \equiv \sum_{i\in V} p(i,d) > 0,\ \forall\ (i,d)$ so that at each time unit exactly one of the pages is requested by one of the users. Define the *comparative cost of an access request for page d with regard to a page d' that is already resident in an NDN node j* as:

$$\begin{aligned}\tau(d,d',j) = \sum_{i\in V} \{&p(i,d')[B(d',i) - T(i,j)] \\ &+ p(i,d)[T(i,j) - B(d,i)]\},\end{aligned} \tag{5.9}$$

This quantity is the average cost of selecting page d' located at j for removal from memory when d is referenced at time t, and it takes into account both the possible gain or loss in comparing the access cost of d and d' with respect to location j, and the chance that d' may be referenced again after it has been removed.

Remark 4 Let us denote by M_t the content of all the memories at the NDN nodes at time t, i.e. $M_t = \cup_{j\in N} M_t(j)$. Since $p_d > 0$, we know that if we are operating "on demand", then there exists a finite time instant $t_0 < \infty$ such that for all $t > t_0 > 0$ we have $|M_t| = m$ where m is the

total memory capacity in number of the network. Let $n = |N|$ be the number of NDN nodes.

Heuristic 1 A useful heuristic for optimising system performance is as follows:

- *If at time t a page fault occurs*, i.e. $r_t = (i, d),\ s.t.\ d \notin M_t$, we select a page d', located at j, for removal from M_t, that has the smallest value of $\tau(d, d', j)$. If there are several such pages, among them we would select one of those that has also the smallest total reference probability $\sum_{i \in V} p(i, d')$.

In the above analysis, the system is constrained to making a replacement when a page fault occurs, which could be suboptimal and consume significant network overhead.

Heuristic 2 Therefore, we present a second heuristic:

- *If at time t a page fault occurs*, i.e. $r_t = (i, d),\ s.t.\ d \notin M_t$, we select a page d', located at j, for removal from M_t, that has the smallest value of $\tau(d, d', j)$ and for which $\tau(d, d', j) \leq 0$. If there are several such pages, among them we would select one of those that has also the smallest total reference probability $\sum_{i \in V} p(i, d')$. Otherwise, d is not cached.

The comparative cost in equation (5.9) can be further generalised as

$$\begin{aligned} \tau(d, d', j) = \sum_{i \in V} \{ & p(i, d')\beta(d')[B(d', i) - T(i, j)] \\ & + p(i, d)\beta(d)[T(i, j) - B(d, i)]\}, \end{aligned} \tag{5.10}$$

where $\beta(d)$ represents the "significance factor" of page d. A network administrator can carefully choose the values of $\beta(d)\ \forall d \in D$ to combine differential services to publishers and customer experience in making caching decisions in the network.

5.4 Cache Management Framework for Information Centric Networks

We now present a cache management framework for implementing on-demand cache management algorithms in ICN-based networks. Our goal is a framework which allows for continuous monitoring of the changing variables and implements caching decisions in a pragmatic way to reduce or increase a given cost. Also, the framework must be adaptable to the different ICN architectures.

We propose placing special nodes, which we call "Cache Management nodes" (CMNs), at the access points of the network as shown in Fig. 5.1. These nodes coordinate to implement the caching policy in real-time. CRs in the network, at regular intervals, send updates of the observed request rates to the CMNs. This information could be sent along with the normal routing updates, therefore reducing the extra communication cost required. Triggered updates can also be sent by piggybacking them on the request packets. Because of their location, CMNs are aware when a cache miss occurs and a decision can then be made when the requested data object returns. The framework requires the CMNs to have a full and accurate view of the cache configurations of the network.

To carry out a replacement decision, the CMN creates a copy of the data object and sends it towards the designated CR along with the identification for the content to be replaced. The original data object is forwarded normally to satisfy the request. On implementing a replacement, a CR will signal the rest of the network to adjust their forwarding tables.

5.4.1 Cache Management for an NDN-based network

In this section, we describe a caching strategy for an NDN-based network which combines the framework in Figure 5.1 and the proposed on-demand cache management algorithms. We refer to this strategy as the *on-demand cache management system ODCMS*, and it can be easily adapted to any ICN-based network.

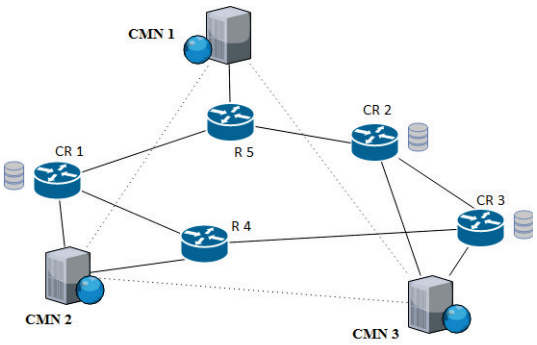


Figure 5.1: A framework for cache management in ICN. The cache management nodes (CMNs) are located at the access points of the network. CMNs coordinate to implement an optimal caching strategy at the cache-enabled routers (CR). The normal store-and-forward routers are labelled R. All the routers that receive external requests send periodic updates about the demand to the CMNs

.

In the NDN implementation, routers are also assigned labels/names which is important for forwarding the signalling/control packets locally. In our implementation, Interest packets are modified for signalling purposes. Signalling Interests are used for sending the periodic access updates to the CMNs and to update the network forwarding tables after each replacement decision. NDN adopts the Type-Length-Value (TLV) encoding scheme to encode its packets such that an NDN packet is a collection of nested-TLVs. So, signalling Interests can be identified by a dedicated type value in the outermost TLV. Signalling Interests are assigned unique names to avoid aggregation and "empty" data packets acting as acknowledgements can be sent to consume them. Each access update information about a content can be encoded into the Interest packets as seperate elements comprising of the content name and its access count TLVs. Furthermore, FIB entries can be installed such that signalling Interests for periodic updates are forwarded towards the nearest CMNs while those to update the forwarding tables after a caching operation are broadcasted. Replacement decisions made by the CMNs are implemented by modifying the unsolicited data pipeline of the NDN communication model [ASZ+14] to permit the forwarding of the data objects to the appropriate CR where it is to be cached. Furthermore, routing faults that may be caused by short-term unsynchronised tables following a replacement are guaranteed to be detected because of the negative acknowledgment feature of NDN [YAM+13].

For the *ODCMS* strategy, the CMNs can derive the distance information and the access weights

or probabilities along with the periodic routing updates in the network. Additional communication overhead will also be incurred to ensure that the CMNs are synchronised and that the forwarding tables match the cache configurations in the network. Regarding the computational complexity at the CMN, a caching decision based on Heuristic 2 in Section 5.3.1 will require a complexity of $\mathcal{O}(m|V|)$. It is important that the computational complexity is independent of the size of the catalogue $|D|$ unlike the approach in [SGFT13].

Finally, monitoring the access rates for all contents available on the Internet will be difficult in practice. Therefore, a network administrator can decide the catalog for which he wants to optimise performance according to the network resources at his disposal.

5.5 Performance Evaluation

In this section, we present extensive evaluations of the proposed caching stategy, ODCMS, on an NDN-based network using the *ndnSim* [MAZ17].

Before outlining how we implemented the ODCMS in these simulations, we first give the following definition. We define a "window" as the period before a CR resends its observed access measurements to the CMN using the signalling Interests, and we set a 5 minutes window length at the CRs to investigate the performance of our system over several windows. Therefore, each CR maintains an access count for each content requested in its current window, and after sending this information to the CMN, it is refreshed. That is, the access count is restarted at the beginning of each window. We expect a network administrator to choose the length of the window that best suits his network system. This choice will largely depend on the resources available in the network and the dynamism in the requests issued by the users of the network. Furthermore, we also incorporate triggered updates to be sent by piggybacking on the Interest packets. We achieve this by computing the $90-th$ percentile (c_{90}) of the access counts at the end of every window. So that when the access count of a particular content reaches a multiple of c_{90}, this information is piggybacked on subsequent Interests guaranteed to reach the CMN. Finally, access counts at the CMN are updated using the smoothed exponential average of each

measurement. Note that an update measurement that arrives on a "normal" Interest will be ignored by the CMN if it is lower than the current estimate.

We compare ODCMS with the following cache management strategies:

- *LCE and LRU* (*LLRU*): Leave copy everywhere combined with LRU replacement policy.

- *Probabilistic caching and LRU* (*PLRU*): Here the decision to cache a content at a node is based on a defined probability.
  We set this probability using the proposed *ProbCache* algorithm [PCP12], which computes the caching probability when a content arrives at a node by weighing the cache capacity along the remainder of the delivery path with the relative position of the router along this path, as shown in equation (5.11).

$$p(x) = \frac{\sum_{i=1}^{c-x+1} N_i}{kN} \cdot \frac{x}{c} \tag{5.11}$$

  where c is the total number of content routers on the path from requester to content source, x is the position of the current node along this path relative to the source, N_i is the cache capacity at node i, N is the average cache capacity along the full path, and k is a constant denoting a target cache capacity along a path.

- Offline optimal placement (*OPT*): Here, since we have a prior knowledge of the popularity ranking of the contents, we first rank the nodes based on their betweenness centrality values and then distribute the most popular contents, the size of the network's capacity, according to this ranking. The betweenness centrality values are computed offline using

$$b(v) = \sum_{s \neq t \neq v} \frac{D_{st}(v)}{D_{st}}, \tag{5.12}$$

  where $b(v)$ is the betweenness centrality value for node v; D_{st} is the number of shortest paths in the network connecting nodes s and t; and $D_{st}(v)$ is the number of these paths

that pass through node v. Table 5.1 shows the betweenness centrality values for each of the nodes in the Elibackbone topology.

The distribution is such that the i-th ranked node r_i caches the data objects ranked $(n(i-1)+1)$, $n(i-1)+2, ..., (n.i)$ in popularity, where n is the number of contents that can be cached in a single node.

Cache placement is fixed and a request is always sent to the nearest node with the required content.

- Label mapping and LRU ($LMLRU$): In this strategy, contents are mapped to the routers via a modulo operation on the content identification.

  In other words, routers are uniquely labelled, and on the arrival of a data object into the network, the content is cached at the router with a matching label. Each node maintains a temporary forwarding table, which is updated in an opportunistic way, to re-route requests that are cached within the network. Ordinarily, requests are forwarded towards the producer or repository.

Table 5.1: Betweenness Centrality values for the Elibackbone topology.

Node	**Betweenness Centrality value**
0	50
1	40
2	206
3	56
4	30
5	0
6	88
7	52
8	14
9	182
10	4
11	46
12	0
13	26
14	256
15	68
16	260
17	36
18	4
19	64

The LLRU, PLRU and LMLRU strategies represent the different categories of caching strategies from the literature that focus primarily on simplicity and increasing content availability instead of optimization. And, the OPT strategy is used as a benchmark for which to compare our solution to a near-optimal one.

For the simulations, we consider an NDN-based network connecting users to content servers. We use the real topologies *Elibackbone* and the *Abilene* according to the datasets provided in [KNF+11]. Every node in each of the networks is configured with the full NDN stack. For simplicity, we consider a single access point for these networks, and therefore, only one CMN is required to serve each network. We set a uniform link speed of $1Gbps$ and $1ms$ delay in the networks.

Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution. At each network node, request arrivals are a Poisson process with an arrival rate of 4 requests per second. For the content catalogue, we consider a content catalog size of $10,000$ objects where each data object is 10KB in size. The link characteristics, together with the chosen request arrival rate and object size, ensure that the network operates below congestion which is one of our simplifying assumptions in Chapter 4. Also, each node in the network can cache a maximum of 10 objects at a time, equivalent to a 10^{-3} cache size to catalogue size ratio.

Finally, for each considered scenario, we run 50 simulations, each lasting 20 minutes simulated time, and we report the average delay measurements for satisfying a request. The simulations for a scenario are randomised by choosing different seeds for the random number generator in each simulation. For the proposed ODCMS, we assume an initial random cache configuration in the network.

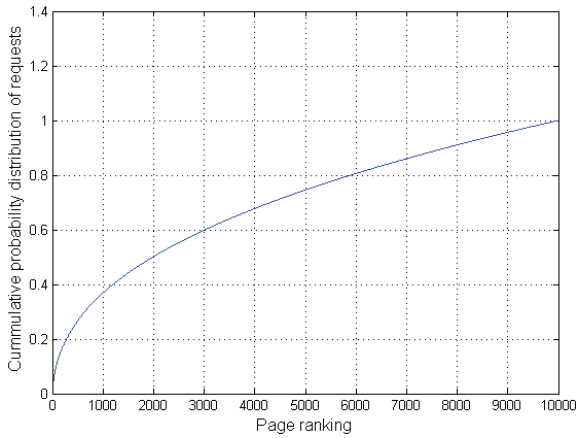


Figure 5.2: MZipf cummulative probability distribution of requests for 10,000 data objects with settings: skew parameter = 0.6, and the plateau parameter = 5.0

5.5.1 Results

Performance evaluation for uniform request patterns

Here, we compare the cache management strategies for scenarios where the demand pattern is uniform across the entire network. We characterise the parameter settings of the MZipf distribution at each node by the proportion, comprising the most popular contents, of the entire catalogue receiving 80% of all the requests. Figure 5.2 shows a plot of the cummulative probability distribution for the MZipf distribution with parameters: $\alpha = 0.6$, $q = 5.0$ and $N = 10000$. The plot shows that the top 6000 most popular contents, which is 60% of the entire catalogue, should account for 80% of all the requests.

Figure 5.3 compares the average delay results for the considered strategies against six different distribution settings for two network topologies. The average delay is measured from the final 10 minutes of each simulation to ensure fair comparison since a cold start at the caches is assumed for the non-optimal strategies.

We can observe from Figure 5.3 that ODCMS achieves significant improvements in comparison to the simple caching strategies (LLRU, PRLU, and LMLRU). Specifically, ODCMS reduces the average delay, for the considered request patterns, by between $11-38\%$ and $15-46\%$ for the Abilene and Elibackbone topologies, respectively, compared to the simple caching strategies. Although, the percentage improvement in delay reduces as the proportion of the most popular

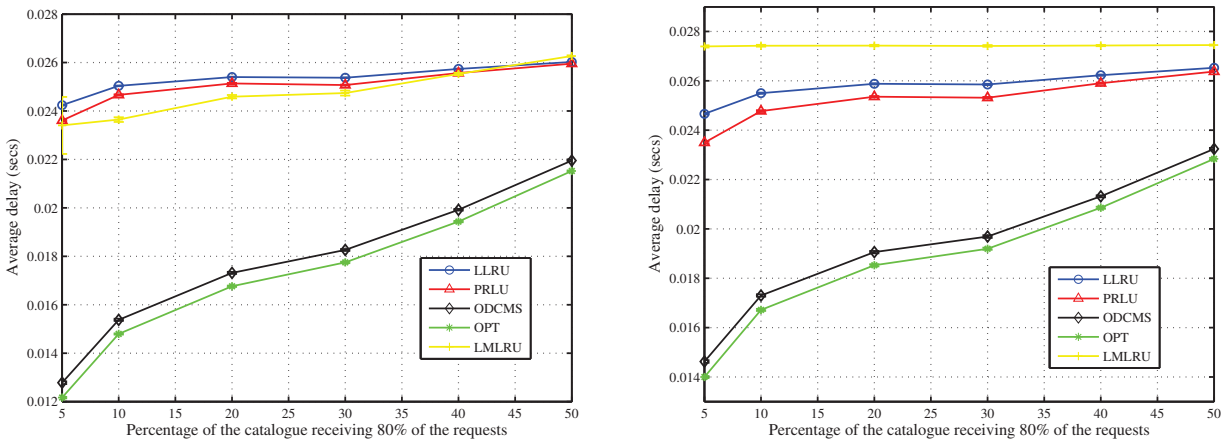


Figure 5.3: Average delay comparison of ODCMS, LLRU, PLRU, OPT and LMLRU under uniform demand patterns for the (**a**) Elibackbone topology and (**b**) Abilene topology. 6 different MZipf demand patterns are considered by varying the proportion of the catalogue that receives 80% of the total requests. N is fixed at 10000 contents. The results reported are the average of 50 randomised simulation runs.

contents, according to our definition, increases. Furthermore, the results show that the ODCMS reaches within 4.8% of the near optimal average delay performance of the OPT strategy when about 5% of the catalogue receive 80% of all the requests for the Abilene topology. As the proportion of the contents receiving most of the requests increases, ODCMS reaches within 2% of the average delay from the OPT strategy. We observe a similar occurrence for the Elibackbone topology.

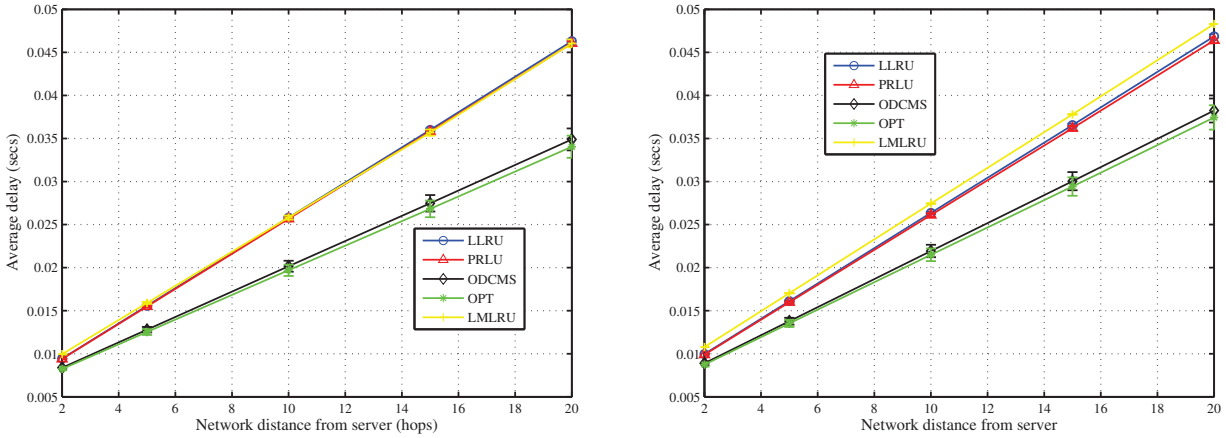
Overall, for a given network cache capacity, as the proportion of the most popular content increases, in-network caching becomes less effective, and the optimal strategies appear to approach the performance of the simple strategies.

Performance evaluation for non-uniform request patterns

We also compare the strategies for non-uniform request patterns by randomly defining the MZipf distribution parameters at each node. The skew parameter at each node is randomly chosen from the range $[0.6, 1.2]$, while the plateau parameter is chosen from the range $[5, 50]$. The catalogue size remains fixed at 10000 contents.

Figure 5.4 compares the average delay results for the considered strategies using two network

topologies while varying the distance of the server or repository from the network. The server distance is defined as the number of hops between the access node of the network and the content server. The average delay is measured starting from the final 10 minutes of each simulation.



(a) (b)

Figure 5.4: Average delay comparison of ODCMS, LLRU, PLRU, OPT and LMLRU under non-uniform demand patterns and at different server distances for the (**a**) Elibackbone topology and (**b**) Abilene topology. The MZipf paramenters at each CR are randomly selected in the ranges: $\alpha \in [0.6, 1.2]; q \in [5, 50]$. N is fixed at 10000 contents. The results reported are the average of 50 randomised simulation runs.

Figure 5.4 indicates that ODCMS can improve the average delay in the networks in comparison to the simple caching strategies. ODCMS reduces the average delay, for the considered server distances, by between $10 - 18\%$ and $10 - 25\%$ for the Abilene and Elibackbone topologies, respectively compared to the simple caching strategies. The percentage improvement increases the further away the content repository is from the network. On the other hand, the results also show that the average network delay achieved by ODCMS remains within 2.5% of the near optimal average delay performance of the OPT strategy for both network topologies and for the considered server distances. Although, this percentage steadily increases as the server distance increases.

Not surprisingly, in general, the closer the content repositories are to the network, the less efficient the optimal methods will be compared to the simple strategies.

In Figure 5.5, we further investigate how ODCMS performs in each window and consider with the case where updates are not triggered. Figure 5.5 shows that for both the OPT strategy

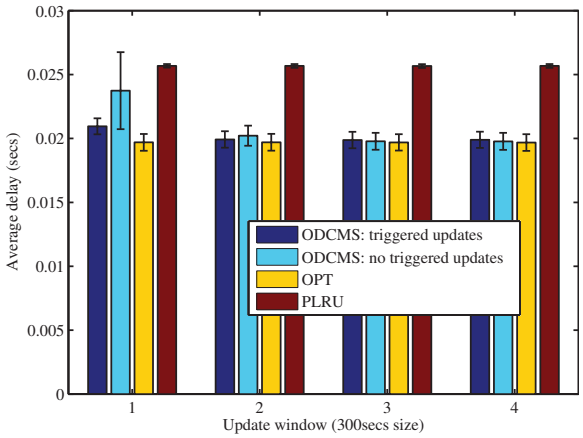


Figure 5.5: Average delay comparison of ODCMS, OPT, and PLRU in each window. For ODCMS, we consider 2 cases: (i) ODCMS with triggered updates and (ii) ODCMS without triggered updates. The size of each window is 300 secs, and we show results for 4 windows. Triggered updates are piggybacked on the "normal" Interests. The results reported are the average of 50 randomised simulation runs.

and the PLRU strategy, there is no significant change in performance across the windows. While OPT produces a near-optimal average delay because of the fixed cache configuration, the PLRU strategy increases the average delay by about 23.3% compared to the OPT strategy in each window. On the other hand, at the end of the fourth window, ODCMS is able to achieve an average delay that is less than 1% above the performance produced by the OPT strategy. Furthermore, by using triggered updates, ODCMS can converge quicker as shown by the average delay at the end of the first window before the periodic updates have been sent. Recall that we set an initial random cache configuration in the network for the ODCMS strategy. Therefore, without triggering updates, there is no optimisation in the first window, and the performance of ODCMS largely depends on the initial cache configuration, hence, the comparatively larger deviation from the mean as shown in Figure 5.5. Eventually, ODCMS without triggered updates fully converges at the end of the third window.

The results we have presented so far suggest the significant benefits a simple cache optimisation system like ODCMS can offer over simple caching strategies. In fact, the LMLRU, which tries to maximise content availability by simply eliminating cache replication without any form of optimisation, produces similar, and in some cases worse, performances compared to the ubiquitous strategies, LLRU and PLRU.

5.6 Conclusion

In this chapter, we presented a simple cache optimisation strategy, the on-demand cache management strategy (ODCMS), for an ICN-based network. Our proposed approach works on-demand such that caching decisions follow the user requests, enabling replacement decisions to be adapted in real-time and in a pragmatic way. We implemented ODCMS specifically on an NDN-based network and showed, via simulations, that it can converge close to a near-optimum solution. Our results also showed the advantages of ODCMS over simple caching strategies that perform no optimisation.

Chapter 6

Interest Forwarding in Named Data Networking Using Reinforcement Learning

6.1 Introduction

The *Named Data Networking* (NDN) [NSF, ZAB+14] is one of the prominent ICN initiatives. In NDN, requests are sent in *Interest packets*, while *Data packets* carry the requested contents; both packets carry the name or identification of the desired data object. An Interest is uniquely identified by its *name* and *nonce* values, where the *nonce* is randomly generated by the requesting application.

An NDN content router (CR) maintains three main data structures for implementing the forwarding plane: a *content store* (CS) which acts as a cache for Data packets, a *pending interest table* (PIT) which keeps track of unanswered requests, and a *forwarding information base* (FIB) which maps reachable name prefixes to outgoing interfaces. The FIB is updated by a name-based routing protocol. An Interest that arrives and matches an entry in the PIT will cause an update of the matching PIT entry whereby the Interest's arrival interface is added to the list of

requesting interfaces. This process is referred to as *Interest aggregation*, and it can be efficient in managing traffic especially for highly popular contents. Furthermore, Interest NACKs were introduced in NDN to address the inefficiencies that result from the dangling states in the PIT caused by unsatisfied Interests. When it cannot forward nor satisfy an Interest, an NDN router responds with an Interest NACK; the Interest NACK also carries a code describing the reasons. Therefore, Interest NACKs can help the network to quickly and informedly detect faults and, when possible, try other forwarding options.

The NDN architecture differs significantly from most of the other ICN proposals because it incorporates an intelligent forwarding plane through its *forwarding strategy module*. The forwarding strategy module is a program, defined for each prefix in the FIB of the CR, which makes decisions for forwarding Interests. It is expected to leverage on both the routing information in the FIB and the observed packet delivery measurements in making adaptive decisions. Furthermore, such an intelligent forwarding plane could relax the stringent convergence and correctness demands commonly placed on the routing layer as explored by the work in [LGZ+16]. An NDN node matches an Interest, using the content name, in the CS, PIT and FIB, in that order. A match in the FIB results in the strategy module being called upon to decide the Interest's next hop. Data packets are sent along the reverse paths used by their corresponding Interest packets, following the states in the PITs. In this chapter, we propose a learning algorithm for the adaptive forwarding of Interests in NDN.

For scalability and practicality reasons, the name-based routing protocol in NDN can only announce and monitor paths leading to the actual publishers and designated repositories, thus leaving the responsibility of exploiting the in-network caching capability, which is vital for delivering the design goals of NDN, entirely to the forwarding strategy. However, most of the proposals for the strategy layer, including the default algorithm adopted by the NDN project, follow a "monitor-the-routing-paths" approach, whereby their forwarding and probing actions are restricted to the interfaces suggested by the routing layer [JST+09, LGZ+16, ASZ+14, YAW+12, GLAMB15]. This approach limits the forwarding strategy from exploiting local caches closer but outside the routing paths. Also, some of the works that search outside the routing paths either ignore Interest NACKs or are not clear about how unsuccessful Interests

affect their algorithms [BM16, CPC+13]. This becomes crucial because Interests are now more susceptible to forwarding loops since there are no guarantees of path correctness. And, from the analysis in [GLAMB15] which showed the possibility of undetected Interest loops occurring in NDN as a result of Interest aggregation, it is clear that forwarding strategies that do not, at least, offer the same guarantees as the routing layer could be disadvantageous.

In this chapter, we develop a dynamic self-aware [Gel17] strategy layer for NDN architectures to offer fast content delivery using local content stores, and we also keep the existing capabilities of the routing layer. The NDN forwarding strategy that we implement exploits the Random Neural Network (RNN) [Gel90b] with Reinforcement Learning, similar to the Cognitive Packet Network (CPN) [Gel09, BCC+17]. An overlay network with a similar scheme is described in [BWG16]. Our results, via simulations, show the following:

i Forwarding strategy algorithms for NDN that do not address how unsuccessful Interests are handled can significantly degrade performance in terms of successful deliveries.

ii Our approach achieves better delivery performance than a strategy that uses fixed paths from the routing layer and a more efficient performance than a strategy that retrieves contents from the nearest possible caches by flooding the requests.

The remainder of this chapter is organised as follows. A review of the relevant literature is presented in Section 6.2. The proposed strategy, *NDN forwarding strategy with Reinforcement Learning using the RNN* is outlined in Section 6.3, followed by the results of our extensive experimentation in Section 6.4. Finally, we conclude the chapter in Section 6.5.

6.2 Related Work

It is shown in [RR14] that coupling caching and forwarding is essential in ICN to significantly benefit from ubiquitous caching. The authors first studied an optimal forwarding policy, the

ideal Nearest Replica Routing (iNRR), in which CRs forward their requests to the nearest possible replica with the help of an "oracle" that keeps track of the network's caching state. Then, due to the cost of realising such an oracle, they proposed practical implementations where CRs periodically explore a given neighborhood in the network by flooding the request. A comprehensive review of caching strategies that have been evaluated in the ICN research is presented in [ZLZ15].

In the NDN context, Jacobson et al. [JST+09] proposed forwarding an Interest along all the interfaces suggested by the routing layer excluding the interface the Interest arrives on, referred to as the *multicast strategy*. The *best route strategy* [ASZ+14] forwards interests using the available upstream with the lowest routing costs. The Adaptive SRTT-based Forwarding Strategy (ASF) is proposed in [LGZ+16]. ASF sends an Interest using the upstream with the lowest measured SRTT among those put forward by the routing protocol. To gather measurements, ASF also probes alternative interfaces at intervals. The current NDN forwarding strategy [ZAB+14, YAW+12, YAM+13] combines interface ranking and colour classification to decide where to forward Interests. Interfaces suggested by the routing protocol are first classified using a colour scheme according to how well they are known to return Data, then the interfaces are ranked in each class using some metric, usually the smoothed round trip time (SRTT). The forwarding logic is to use the highest ranked available interface in the best possible classification. NDN also introduced negative acknowledgements, referred to as Interest NACKs, to address the inefficiencies that result from the dangling states in the PIT caused by unsatisfied Interests. When it cannot forward nor satisfy an Interest, a CR responds with an Interest NACK; the Interest NACK also carries a code describing the reasons.

To eliminate undetected loops in NDN, the *Strategy for Interest Forwarding and Aggregation with Hop-Counts* (SIFAH) is proposed in [GLAMB15]. In SIFAH, a CR accepts to forward an Interest only if there exists, based on distance information to the content source, a neighbour node that strictly moves the packet closer to the content. Otherwise, an Interest NACK is returned. The distance information used in SIFAH is the number of hops to the content repository, and it is provided to the forwarding plane by the routing protocol. Also, by replacing nonce values with distance information, SIFAH reduces the memory overhead incurred by the

PIT. While SIFAH guarantees a correct forwarding strategy, it achieves this by limiting the dynamism of the forwarding plane in making adaptive decisions. In addition, the conditions it uses for loop detection, and therefore for forwarding Interests, are sufficient conditions. This means that an interface can fail these conditions and yet not lead to an Interest loop being formed. Hence, the possibility of unnecessarily denying service to Interests exists.

Another class of forwarding strategies referred to as the *multipath* strategies [QRWM13, NFST15, PRH17], dynamically assign a forwarding weight or probability to each interface of a CR, which determines the proportion of Interest traffic sent on an interface. The main aim here is to achieve good load balancing and manage congestion in the network.

In our work, we propose implementing the strategy module of the NDN architecture using an online learning algorithm. Reinforcement learning has been previously proposed for the NDN forwarding strategy. In [BM16], the *multi-armed bandits strategy* (MABS) is developed which assumes no knowledge of path information from the routing layer. As a result, when no forwarding information is available, the CR floods a request on all its interfaces. *INFORM*, presented in [CPC+13] and inspired by the Q-routing algorithm [BL93], adopts a more similar approach to ours by leveraging on the routing layer. INFORM alternates between exploration and exploitation phases when making forwarding decisions. In the initial exploration phase when no learning has occurred, a received Interest is sent using the best interface according to the information in the FIB, and a copy of the Interest is also sent on a randomly selected interface. The same actions are repeated in subsequent exploration phases except that the best interface will be the one learnt by the algorithm. Only the best interface computed after an exploration phase is used during the following exploitation phase. In these methods, the authors do not address the handling of Interest NACKs, which becomes significant because the exploration of the network, necessary for the learning algorithm, increases the possibilities of Interest loops.

Our approach is inspired from the CPN routing protocol which, in most of its implementations, employs a reinforcement learning algorithm using the RNN [Gel90b] to establish and maintain connections between network nodes. The algorithm used has been shown to possess good

convergence properties during an initial learning phase and good sensitivity to environmental changes [Hal00, DG14], hence its success in CPN routing. A comprehensive review of the variations, applications, and performance evaluations of the CPN can be found in [Sak10].

6.3 NDN Forwarding with Reinforcement Learning using the RNN

We now present NDNFS-RLRNN that uses reinforcement learning (RL) with the RNN for the strategy module per prefix in the NDN architecture as detailed in Chapter 3, which operates with a form of smart Opportunistic Communication [GG11], supported by a name-based routing protocol with online measurements from the state recorded in the PIT. Our goal is to exploit the convergence properties of the RL with RNN algorithm which have been well studied in [Hal00, DG14] algorithm to effectively search for local content store hits outside the routing paths. Unlike similar approaches, we address the performance degradation that could be introduced by unsuccessful Interests as a result of forwarding Interests using interfaces not suggested in the routing table.

In NDNFS-RLRNN, an RNN is created for a prefix in the FIB and its creation is triggered by a new Interest for the corresponding prefix. In its initial state, the RNN only knows of the routing preferences for its prefix and is yet to be updated by packet delivery measurements. Here, NDNFS-RLRNN's forwarding decision will be to use the best interface according to the routing layer. Each RNN per prefix at the CR has as many neurons as the number k of outgoing links of the router, and the RNN has k^2 weights, and if the router has n_D active destinations, the router will have a total of n_D RNNs to handle each of the destinations, or a total of $n_D \times k^2$ entries. A conventional routing table at the NDN router will have $n_S \times n_D \times k$ entries for n_S active sources. Thus when $n_S > k$, our scheme uses a smaller data structure per router than a conventional NDN scheme [GK14].

On the arrival of a Data packet, a reward value is computed for the arrival interface and used

to update the RNN as illustrated in the RLRNN algorithm in Section 3.3 of Chapter 3. The goal of our distributed reinforcement learning is to minimise the delay for retrieving contents, so we estimate reward using the RTT values. Let ${T^j}_I$ be the time an NDN router forwards an Interest along an interface j and ${T^j}_D$ be the arrival time of the Data packet satisfying the Interest which also arrives on the same interface, we can estimate the reward, R^j as the inverse of the RTT using

$$R^j = ({T^j}_D - {T^j}_I)^{-1} \tag{6.1}$$

For an updated RNN, the forwarding decision of NDNFS-RLRNN is, excluding the arrival interface, with a high probability, p, only the interface corresponding to the most excited neuron, which is regarded as the best interface, is used or with probability $(1 - p)$ a probing decision is made. Probing involves sending the original Interest along the best-known interface and a copy of the Interest on a randomly selected interface for exploration. We refer to p as the *probe parameter*. The Interest and Data forwarding processes of NDNFS-RLRNN are summarised using the flowcharts in Figures 6.1 and 6.2, respectively.

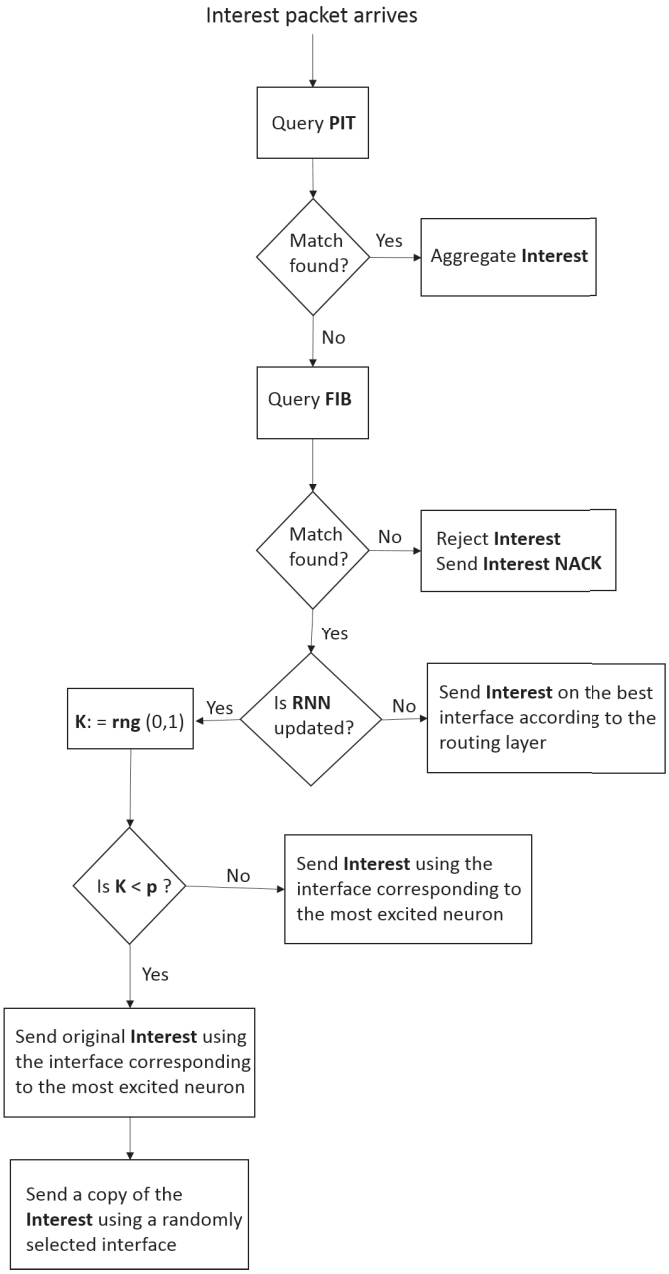


Figure 6.1: Interest forwarding and the probing process of NDNFS-RLRNN. It uses the ϵ-greedy approach for the probing. $rng(0,1)$ is a random number generator that returns a number in the range $[0, 1)$, and p is the probe parameter of the RNN.

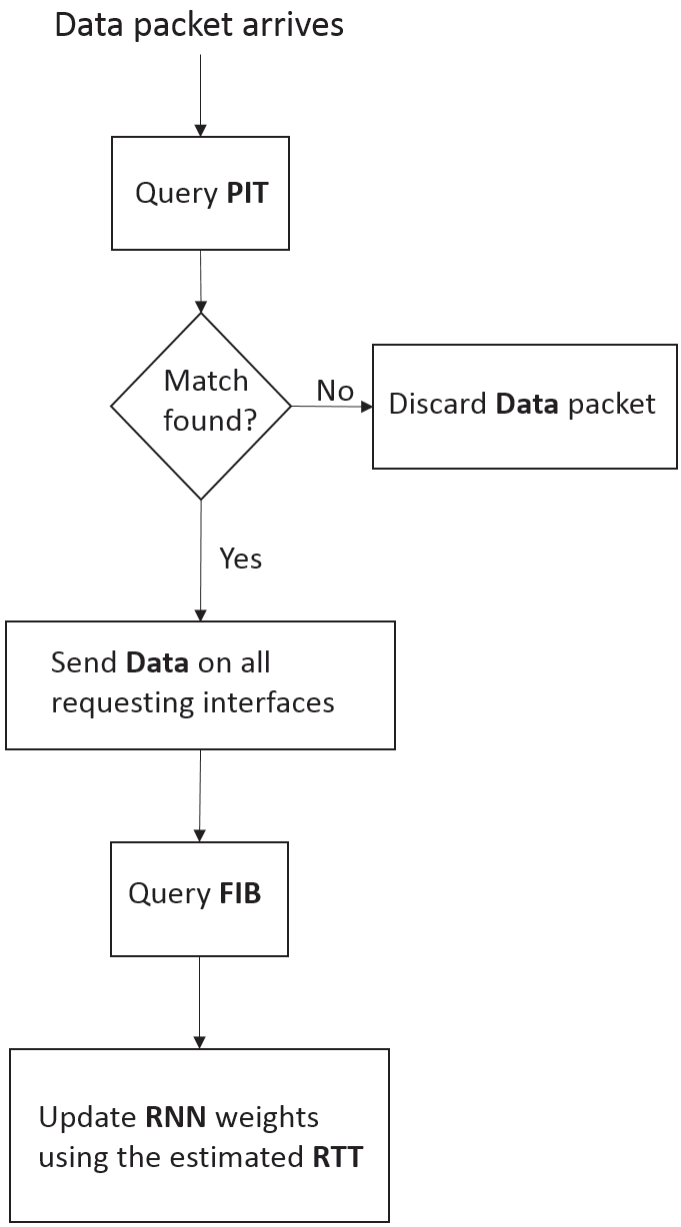


Figure 6.2: Data forwarding in NDNFS-RLRNN. After sending the Data on all the requesting interfaces, the estimated RTT is used to update the weights of the corresponding RNN.

6.3.1 Addressing Correctness in NDNFS-RLRNN

In this section, we modify the NDNFS-RLRNN forwarding design we have just described to address issues of unsuccessful Interests and match the routing layer's guarantees for content retrieval. Before detailing the modifications, we first describe a possible scenario where a loop can go undetected in NDN based on the analysis first carried out in [GLAMB15].

Consider the network segment in Figure 6.3 consisting of 5 NDN content routers (A - E) and a publisher node J.

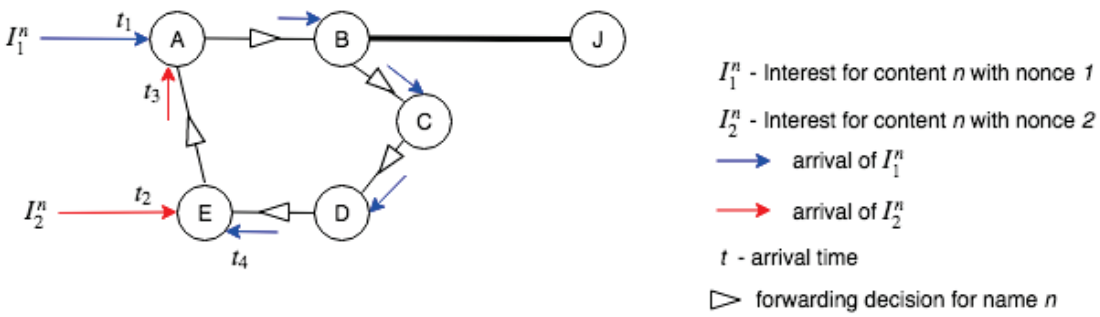


Figure 6.3: Interest loop in NDN.

Let, for content n, the forwarding tables at the content routers form an Interest loop as shown by the arrowheads on the links in Figure 6.3. In other words, an Interest for content n that arrives at any of the CRs will travel around the loop consisting of content routers A - E. Ordinarily, the loop detection mechanism of NDN which utilises the uniqueness of each Interest will successfully detect an Interest loop once an Interest completes a trip around this loop. However, consider the situation depicted in Figure 6.3 where Interest ${I_1}^n$ arrives at content router A at time t_1 and Interest ${I_2}^n$ arrives content router E at t_2. Assuming that $t_x < t_y$ if $x < y$, ${I_2}^n$ will be aggregaged by A at time t_3 and ${I_1}^n$ by E at time t_4, thereby forming a loop that would not be detected. In fact, resending these Interests from downstream nodes will not solve the problem. Although, the loop could be bypassed if other paths exist to the publisher which do not include CRs A - E. Only the expiration of the PIT entries at the CRs for the content n will break the loop.

Now let the PIT entry for ${I_1}^n$ in CR B expire and, as a result, it sends an Interest NACK to the only requesting interface which leads to CR A. CR A will, in turn, send Interest NACKs to the arrival links for ${I_1}^n$ and ${I_2}^n$, thereby breaking the Interest loop. Nevertheless, if the NDN forwarding strategy does not change its state as a result of receiving an Interest NACK, this Interest loop for content n will persist in the forwarding tables.

Our initial assumption of a forwarding loop in the CRs in Figure 6.3 is justified, especially for forwarding strategies that randomly select links for sending Interests. Consider the initial

working situation where the forwarding tables at the CRs all point towards the publisher J. The width of the link between CR B and node J in Figure 6.3 is larger compared to the other links to indicate that is has a significantly longer transit time. Let n become cached in CR E and the forwarding algorithms at CRs B, C, and D, through exploration, have discovered the closer cache at E for the content and independently updated their forwarding tables. Then, if the content n is CR E is evicted in this state, the forwarding tables will form the exact loop shown in Figure 6.3.

Therefore, we present a modified NDNFS-RLRNN (mNDNFS-RLRNN) in which we address the possible correctness issues in the initial design. The modification includes

- refining the exploration or probing process,
- relaxing the Interest aggregation principle of NDN,
- handling unsuccessful Interests.

We first introduce the idea of marking an Interest packet for probing. This means that a router identifies an Interest as either a "normal" Interest or a probe Interest. The probing process now involves sending 2 different Interest packets: a "norma" Interest and a copy which is sent as a probe Interest. A "normal" Interest is always sent using the best interface put forward by the routing protocol. Furthermore, when a match is not found in the PIT, probing is triggered only in two situations. Firstly, when a "normal" Interest arrives and the best face according to the RNN is also in the set of interfaces being suggested in the FIB. Here, a probe Interest is also sent on a random interface based on probability p. Secondly, when a "normal" Interest arrives but the best face according to the RNN is not in the set of interfaces being suggested in the FIB. In this case, a probe Interest is sent on the RNN interface while the "normal" Interest is sent on the best interface in the FIB.

We also adjust Interest aggregation such that when a "normal" Interest arrives at a CR, and there is a match in the PIT, in addition to updating the list of requesting interfaces, the Interest is sent using the best interface in the FIB. For probe Interests, aggregation occurs

without forwarding the Interest. This relaxation of Interest aggregation together with the refined exploration process ensures that the assurances from the routing layer remain at our forwarding plane. Also, it is clear under these rules that the forwarding loop illustrated in Figure 6.3 will not prevent an Interest from reaching the publisher J because CR B will be forced to send the "normal" Interest to J and a probe Interest to CR C. Figure 6.4 summarises the Interest forwarding and probing process of mNDNFS-RLRNN.

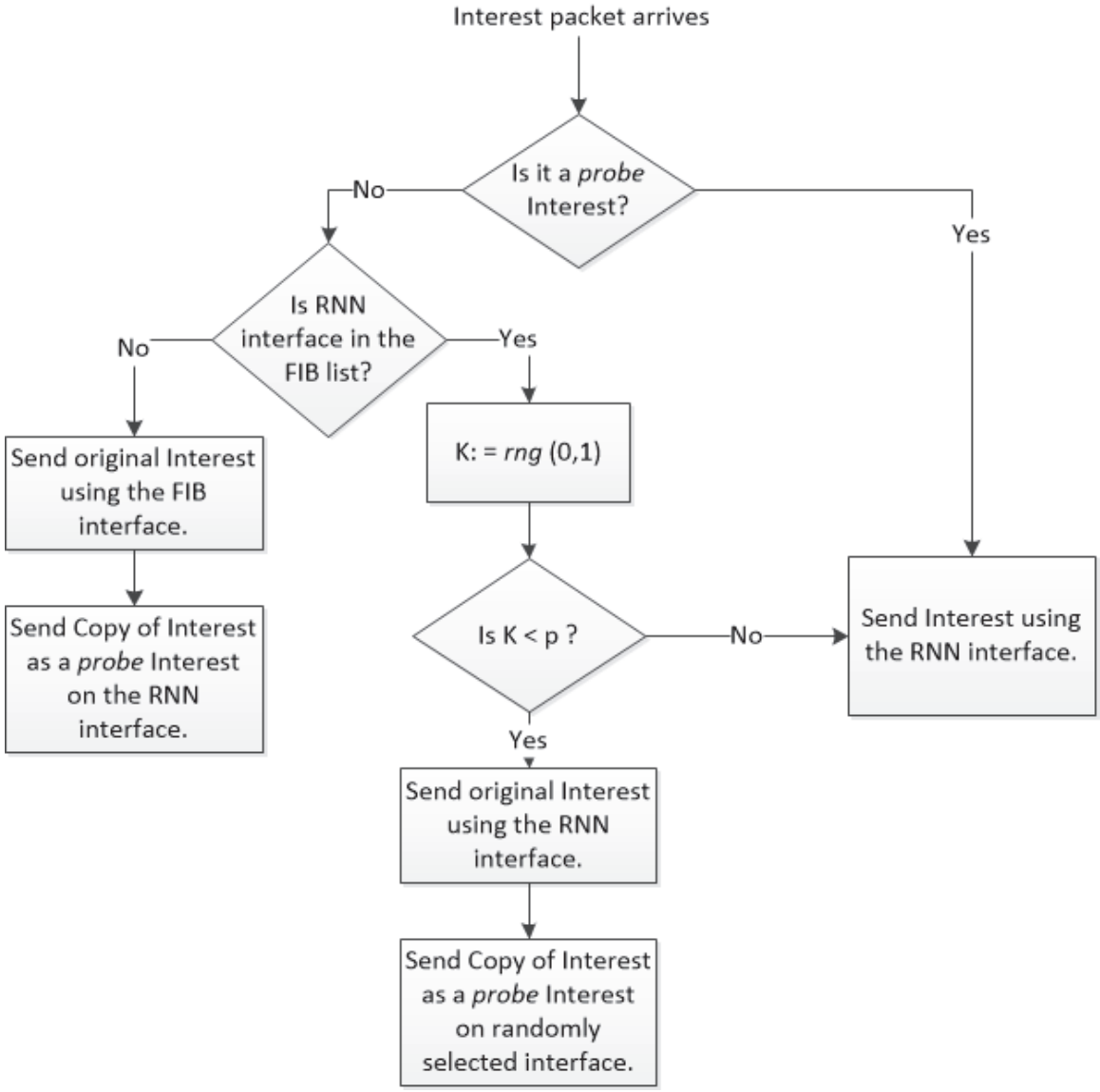


Figure 6.4: Interest forwarding and probing process of the modified NDNFS-RLRNN. $rng(0,1)$ is a random number generator that returns a number in the range $[0, 1)$. p is the probe parameter of the RNN. The RNN interface represents the best interface learned by the proposed algorithm, and the FIB interface is the best interface announced by the routing protocol.

Finally, we have also adopted the use of Interest NACKS as in the current NDN in our approach.

When an Interest NACK is returned on the best interface according to the learning algorithm, the learning algorithm is forced to return to its initial state where it follows the FIB preferences. This is because we recognise that the volatile nature of the local caches means that convergence is not always possible to reach and could come at the cost of unnecessarily denying service to Interests. In other words, our algorithm seeks convergence only when it would not be detrimental to performance. In addition, a CR can resend an Interest if it receives Interest NACKs on all the interfaces in its outgoing list if there still exists unused paths according to the FIB. Otherwise, Interest NACKs are returned on all the requesting interfaces.

To reduce the inefficiencies that could be introduced by out-of-date information and for efficient use of resources, when an RNN receives no feedback for τ_r time units, it is deleted. We refer to τ_r as the *refresh time*. As a result, an RNN will remain in the system only if it is considered active according to the above refresh time condition. This is necessary to manage the limited resources at each CR. Furthermore, a limit could be set for the number of RNNs at a CR depending on the available network resources, such that new Interests that arrive after this limit is reached will be sent using the routing layer options without any attempt to search locally.

6.4 Performance Evaluation

In this section, we present initial evaluations of the performance of NDNFS-RLRNN through extensive simulations using the *ndnSim* [MAZ17], an NS-3-based simulator which already exists for NDN.

For the simulations, we consider an NDN-based network connecting users to content servers. We use the real topologies *Elibackbone* and the *Geant* according to the datasets provided in [KNF+11]. We first examine a network with a single access router through which requests are served from the content servers using the Elibackbone topology. Using the Geant topology, we then simulate a flat network with multiple publishers. At each node, except the access node and the publisher nodes, external request arrivals from the consumers are a Poisson process with a

rate of 5 request packets per second, and each data object is 10KB in size. All the simulations begin with cold or empty caches at the nodes. For these tests, the ndnSim simulator did not scale well, in terms of the available memory [SFP17], with the size of the content catalogue because each content requires a separate entry in the FIB in each node. As a result, we consider catalogue sizes in the range [1000, 5000].

Furthermore, we first investigate the improvements of mNDNFS-RLRNN before comparing our approach with two other forwarding strategies: the *Adaptive SRTT-based Forwarding Strategy* (ASF) [LGZ+16], and a *Nearest Replica Routing* (NRR) strategy [RR14]. The *ndnSim* has the ASF algorithm pre-installed. The ASF strategy uses the upstream with the lowest measured SRTT and probes alternative interfaces suggested in the FIB at intervals. In other words, in the ASF strategy, forwarding decisions are restricted to the routing layer preferences. The length of the probing interval is reduced from the default value of 60 secs to 3 secs to increase probing, and we install all possible routes in the FIB such that no loop exists in the forwarding tables. For the NRR strategy, we implement a multicast algorithm that sends each request on all the interfaces of a node except the arrival interface if there is no match in the PIT. If there is a PIT match, the Interest is aggregated and not sent. This strategy guarantees that the requests, when successful, are satisfied from the closest caches. Other approaches for nearest replica routing are possible where the flooding is periodic as suggested in [RR14], but we have chosen this simple method which avoids any parameter settings. The forwarding strategies are implemented per content in the catalogue. We also set Interest retransmissions at each consumer using the smoothed round trip time (SRTT) estimates based on the illustration in [rfc81]. In other words, if a request is not satisfied within an interval whose length is the current SRTT estimate, the Interest is resent. The algorithm for updating the SRTT estimates comes with the ndnSim version 2.5 that we have used.

Since the cache states in the network will influence the performance, the forwarding strategies are compared under three different caching policies from the literature:

- Leave copy everywhere combined with LRU replacement policy (*LCE*): Here, the CRs cache every data packet received.

- Betweenness centrality policy and LRU (*Betw*): In *Betw*, proposed in [CHPP13], only the routers with the highest betweenness centrality values along the delivery path cache the content. These values are computed offline using the betweenness centrality definition in graph theory.

- *ProbCache and LRU*: Here the decision to cache a content at a CR is based on a defined probability. We set this probability using the proposed *ProbCache* policy [PCP12], which computes the caching probability when a content arrives at a node by weighing the cache capacity along the remainder of the delivery path with the relative position of the router along this path, as shown in Equation (6.2).

$$p(x) = \frac{\sum_{i=1}^{c-x+1} N_i}{kN} \cdot \frac{x}{c} \tag{6.2}$$

  where c is the total number of content routers on the path from requester to content source, x is the position of the current node along this path, N_i is the cache capacity at node i, N is the average cache capacity along the full path, and k is a constant denoting a target cache capacity along a path.

The *LCE* and *ProbCache* caching policies are ubiquitous caching policies differentiated by their cache eviction rates. By ubiquitous, we mean that nodes in the network are not prioritised in any way when caching. The *Betw* policy, on the other hand, tries to cache predominantly at the most "central" nodes in the network. These policies represent three categories of on-path caching strategies in the literature.

We compare the performances of the forwarding strategies using the following metrics:

- Cache hit rate: This is measured as the proportion of the requests arriving into the network which are satisfied from the local caches.

- Total successful deliveries: As a measure of the throughput of the system, we count the

total number of Data packets successfully delivered to the consumers.

- Network load or overhead: This is the total number of hops traversed by the network packets, which includes Interests, Data packets and Interest NACKs, per request sent into the network.

Finally, we run each simulation scenario for 400 secs, and all metrics are obtained using the mean and standard deviation of 30 randomised simulation runs. The simulations for a scenario are randomised by choosing different seeds for the random number generator in each simulation. Also, with the chosen arrival rate and the Elibackbone topology, the average number of requests arriving into the network for a 400-second simulation run will be about 38000 requests. Therefore, our choice of the duration of each simulation is reasonable to evaluate performance since the network cache capacity is about 190 objects.

6.4.1 Modification of the NDNFS-RLRNN

We begin our evaluations by investigating the effect of the correctness measures added to our proposed algorithm using the refresh time (τ_r). The length of the refresh time signifies how long before an RNN is considered "outdated" if it receives no new information. A short refresh time can impede the optimisation of the learning algorithm. On the other hand, the volatility of the content stores means that a large value for τ_r increases the possibilities of forwarding Interests based on outdated information, which can increase the number of unsuccessful deliveries and the number of hops used by the Data packets.

In these simulations, we use the Elibackbone topology and consider a uniform content popularity model across the network with Mzipf parameters set at $\alpha = 1.0$ and $q = 5.0$. By uniform popularity, we mean that request arrivals at each CR in the network are characterised by the same MZipf parameters. We fix the catalogue size at 1000 data objects, and each CR is equipped with a cache size of 1% of the catalogue. We compare NDNFS-RLRNN and mNDNFS-RLRNN with $\tau_r \in \{5s, 10s, 20s, 300s\}$ and the probe parameter fixed at $p = 0.3$.

Figure 6.5 compares the total successful Data deliveries at the consumers for NDNFS-RLRNN and mNDNFS-RLRNN under different caching policies at the local content stores. We observe that across all the caching policies, mNDNFS-RLRNN significantly improves the delivery. Due to the modifications we introduced, mNDNFS-RLRNN seems to match the guarantees of the routing layer since the duration of the refresh time does not adversely affect the delivery performance. In contrast, without the modifications, NDNFS-RLRNN forwarding is not correct. In fact, the number of successful deliveries reduces as the refresh time is increased. For example, with the LCE caching policy, NDNFS-RLRNN satisfies about 41% less Interests than mNDNFS-RLRNN at $\tau_r = 5$ secs and 73% less Interests when the refresh time increases to $\tau_r = 300$ secs.

The results confirm that the forwarding strategies that seek convergence by exploring outside the routing paths can suffer from correctness issues because of the volatility of the content stores. Our approach seeks convergence when it is possible but falls back to the routing layer when the network changes too quickly.

Furthermore, we report the effect of the refresh time on the cache hit rate of mNDNFS-RLRNN in Figure 6.6.

We observe that for the LCE caching policy increasing the refresh time reduces the cache hit rate. On the other hand, for both *Betw* and *ProbCache* policies, increasing the refresh time does not adversely affect the cache hit rate performance. Although, for *ProbCache* policy, the cache hit rate at $\tau_r = 20$secs is slightly better than at $\tau_r = 300$secs. This suggests that when the network cache state is very volatile, as is the case when the LCE policy is used, large refresh times can deteriorate the performance of NDNFS-RLRNN because of outdated information.

For the remainder of the analysis, we use NDNFS-RLRNN to refer to the correct and modified version. We also fix the refresh time, τ_r, at 10 seconds for the rest of our simulations.

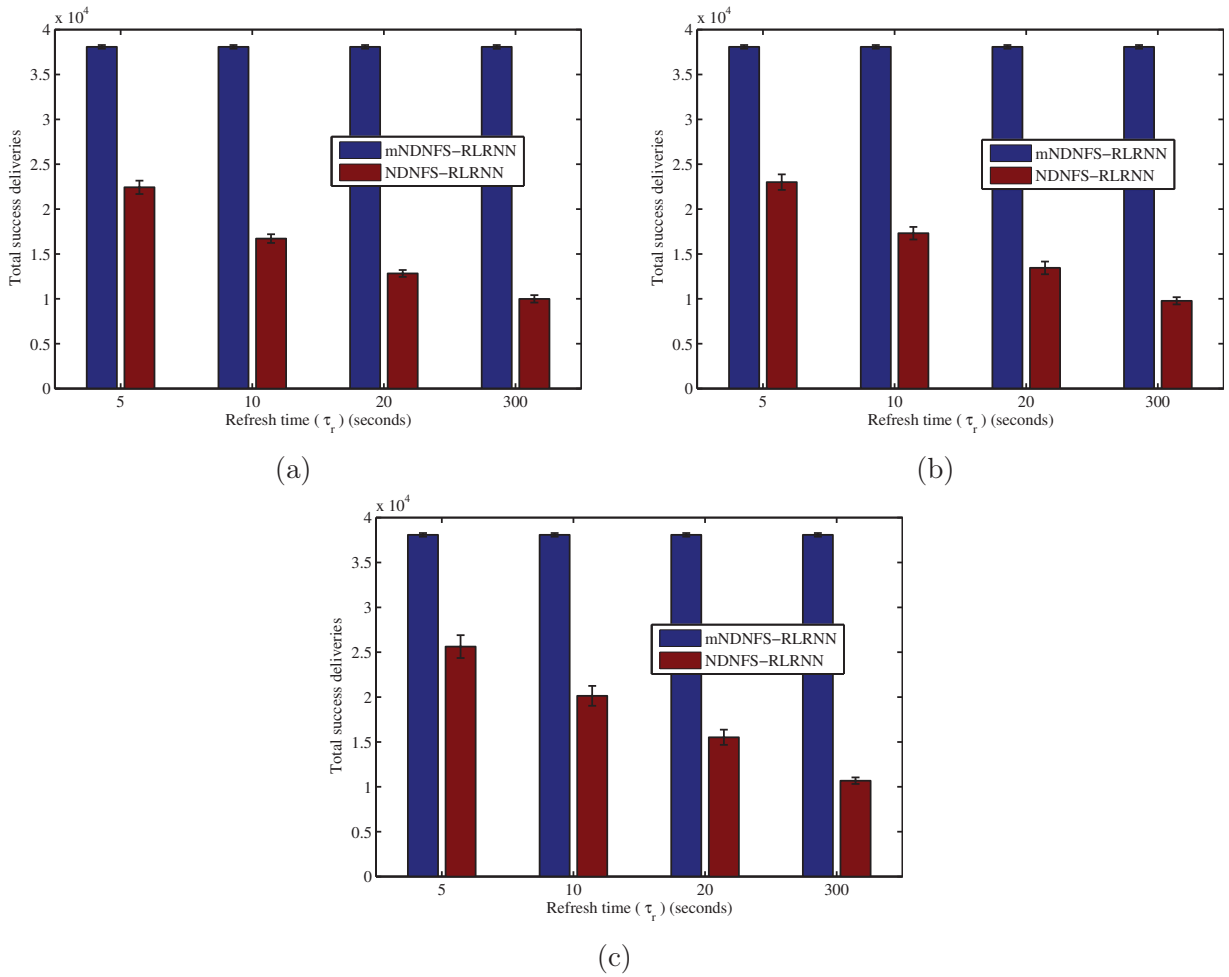


Figure 6.5: Total successful deliveries at the consumers under different on-path caching policies. (**a**) LCE; (**b**) *Betw* and (**c**) *ProbCache*. The plot compares the forwarding stategies, NDNFS-RLRNN and mNDNFSRLRNN with $\tau_r \in \{5s, 10s, 20s, 300s\}$ and the probe parameter fixed at $p = 0.3$. Each CR is equipped with a cache size of 1% of the content catalogue. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings: $q = 5.0, \alpha = 1.0, N = 1000$. The results reported are the average of 30 randomised simulation runs.

6.4.2 Effect of the probe parameter

We now investigate the impact of the probe parameter (p) of the NDNFS-RLRNN algorithm. We retain the same scenario as in our first simulations but vary the probe parameter from 0 to 1. The value of the probe determines how often new paths are explored at each of the CRs. A probe parameter setting of $p = 0$ represents the extreme case where no exploration occurs, and the NDNFS-RLRNN's decision will always be the same as the routing layer's preferred forwarding interface since the corresponding neuron will continue to be reinforced indefinitely. On the other hand, the extreme case of a $p = 1$ configuration will result in the maximum

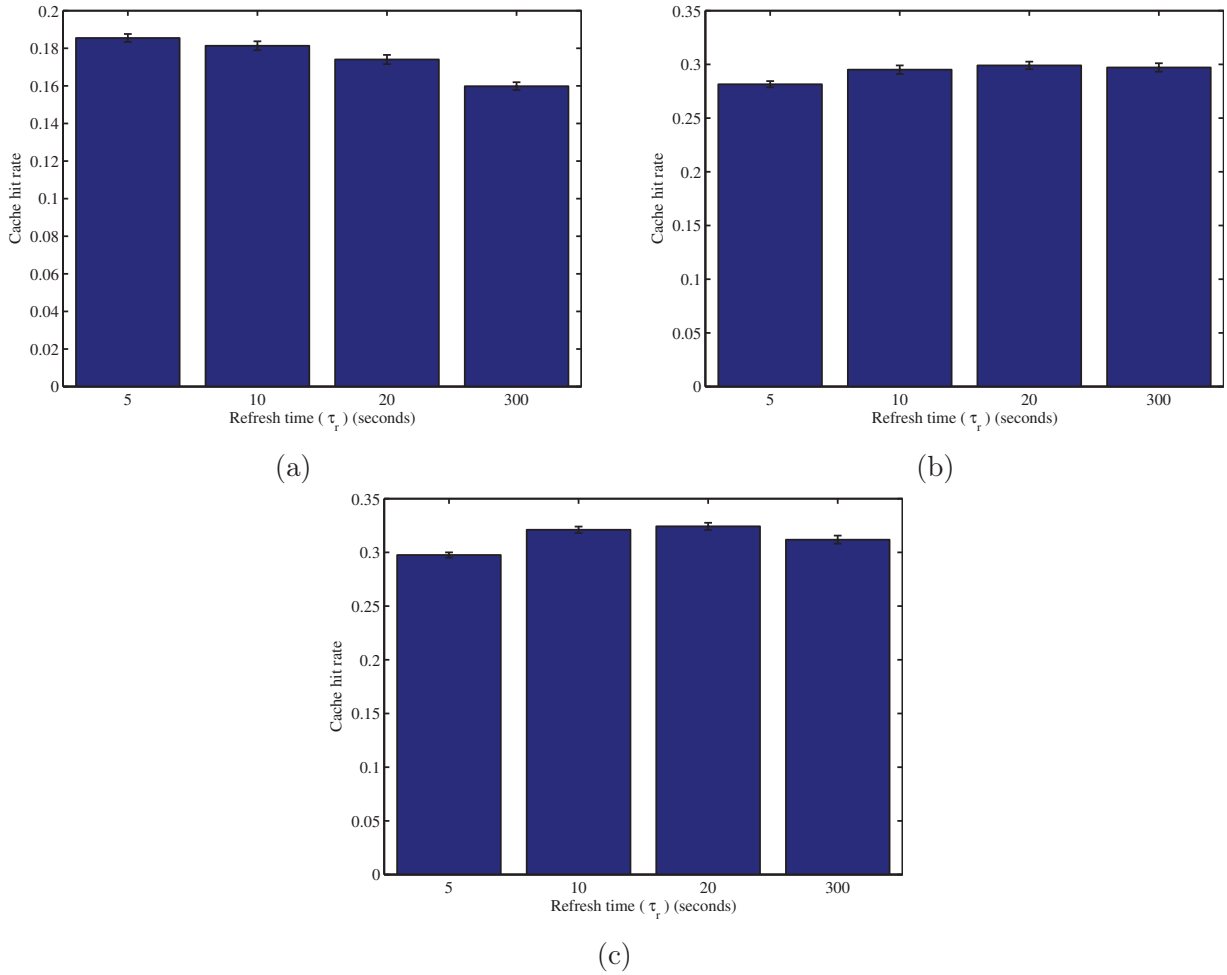
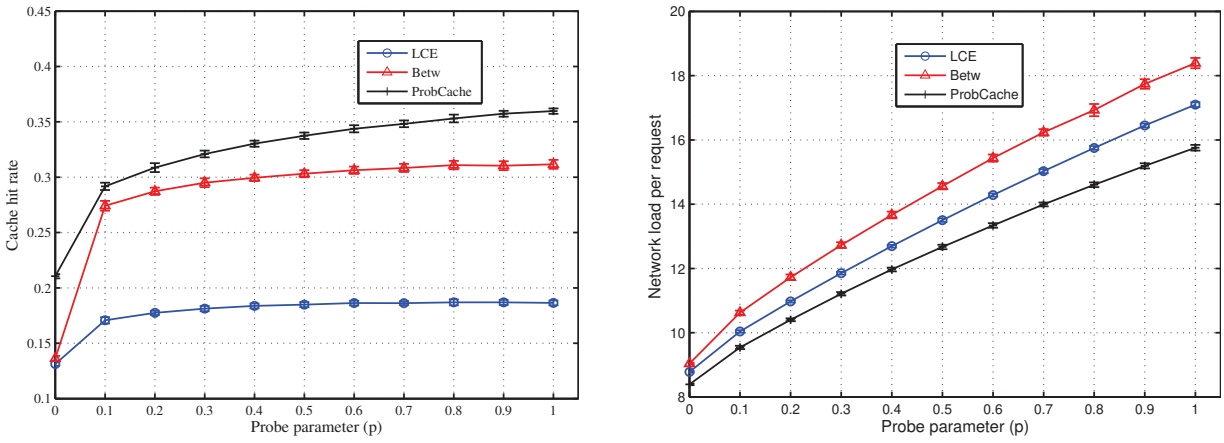


Figure 6.6: Cache hit rate performance of the modified NDNFS-RLRNN for refresh time values $\tau_r \in \{5s, 10s, 20s, 300s\}$. (**a**) LCE; (**b**) *Betw* and (**c**) *ProbCache*. Each CR is equipped with a cache size of 1% of the content catalogue. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings: $q = 5.0, \alpha = 1.0, N = 1000$. The results reported are the average of 30 randomised simulation runs.

possible exploration effort by the algorithm since the CR will always try to send 2 Interests upon receiving a fresh one. Figure 6.7 shows the effect of increasing the probe parameter on both the cache hit rate and the network overhead.

In Figure 6.7, we observe the most significant improvements in the cache hit rate just from increasing p from 0 to 0.1 across all the caching policies. Specifically, increasing the probe parameter from 0 to 0.1 results in a 30%, 101% and 38% increase in the cache hit rate under the LCE policy, *Betw* and *ProbCache* policy, respectively. However, these improvements come at the cost of increasing the network overhead by 14%, 18% and 14% under the LCE policy, *Betw* policy and *ProbCache* policy, respectively. Figure 6.7 also shows that for all the caching policies,



(a) (b)

Figure 6.7: The impact of the probe parameter (p) of the NDNFS-RLRNN forwarding strategy under different on-path caching policies. (**a**) Cache hit rate; (**b**) Network load per request. Each CR is equipped with a cache size of 1% of the content catalogue. The content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings: $\alpha = 1.0$ and $q = 5.0$. The results reported are the average of 30 randomised simulation runs.

increasing p beyond 0.1 produces moderate improvements in the cache hit rate compared to the initial improvement. In fact, the overall improvements from increasing p from 0.1 to 1.0 are 9%, 14% and 23% under the LCE policy, *Betw* policy and *ProbCache* policy, respectively. These results suggest that exploration and the intelligence introduced by the RNN can be beneficial when in-network caching is effective.

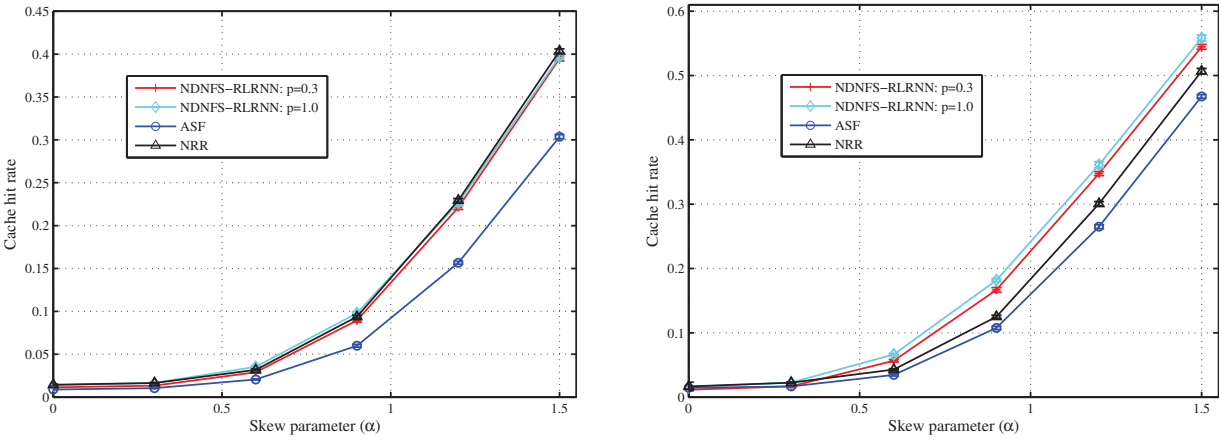
Overall, Figure 6.7a shows that the *ProbCache* policy produces the best cache hit performance, followed by the *Betw* policy, because it combines better redundancy and eviction properties compared with the other policies. The LCE policy produces the worst performance in cache hit rate most likely because of its characteristic high redundancy and high eviction rate. Although the *Betw* policy reduces redundancy compared to LCE, there will be high eviction rates at the most "central" nodes, so allocating cache resources according to the centrality measure could further improve performance. As expected, Figure 6.7b shows that the network overhead increases as the probe parameter increases.

6.4.3 Content Popularity

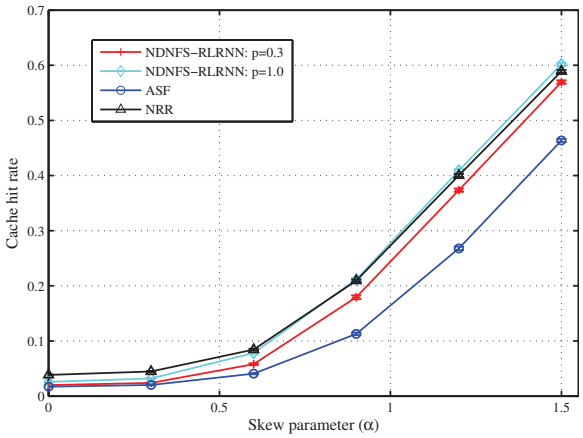
Maintaining a uniform request pattern across the network, we show the impact of the content popularity model by considering skew parameter values (α) of 0, 0.3, 0.6, 0.9, 1.2 and 1.5. The q parameter remains fixed at 5.0. Furthermore, we run the simulations for 2 different NDNFS-RLRNN configurations: NDNFS-RLRNN with $p = 0.3$ for low exploration, and NDNFS-RLRNN with $p = 1.0$ for maximum exploration. Figure 6.8 shows that at $\alpha = 0$, which corresponds to equally popular items, the forwarding strategies produce an identical cache hit rate performances across all the caching policies. This is expected since in-network caching becomes less effective as the proportion of the most popular contents increases for a given network cache capacity. However, as most of the requests concentrate on a smaller proportion of the content catalogue, that is, as α as increases, the forwarding strategies can hit more local caches. As α is increased from 0.6 to 1.5, Figure 6.8 shows that with low exploration NDNFS-RLRNN achieves an improvement in the cache hit rate compared to the ASF strategy of between $30\% - 50\%$, $16\% - 64\%$ and $22\% - 42\%$ under the LCE policy, *Betw* policy and *ProbCache* policy, respectively. We point out that these improvements reduce in percentage as α increases towards 1.5.

Figure 6.8a shows that under the LCE policy, the performances of the two NDNFS-RLRNN algorithms closely follow that of the NRR strategy. Specifically, for the considered skew parameters, while NDNFS-RLRNN with low exploration does not improve upon the cache hit rates produced by the NRR strategy, with full exploration, we only see an improvement of about 11% and 4% for $\alpha = 0.6$ and $\alpha = 0.9$, respectively. We observe similar behaviour in Figure 6.8c under the *ProbCache* where with full exploration, NDNFS-RLRNN can match the cache hit performance of the NRR strategy as α increases.

As a measure of efficiency, we also report the network overhead in Figure 6.9. We observe that for the deterministic forwarding strategies, ASF and NRR, the network overhead reduces as the skew parameter is increased. While for our proposed NDNFS-RLRNN strategy, the overhead appears to initially rise to a maximum value (which occurs between $\alpha = 0.9$ and $\alpha = 1.2$) before falling as α is further increased. Although, for the considered α values, the network



(a) LCE (b) *Betw*



(c) *ProbCache*

Figure 6.8: Cache hit rate performance of NDNFS-RLRNN ($p = 0.3$), ASF and NRR with uniform content popularity distributions, under: (**a**) LCE caching policy, (**b**) *Betw* caching policy and (**c**) *ProbCache* caching policy. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings: $q = 5.0, \alpha = 0.0, 0.3, 0.6, 0.9, 1.2$ and 1.5, and $N = 2000$. Each CR is equipped with a cache size of 0.4% of the content catalogue. The results reported are the average of 30 randomised simulation runs.

overhead for the NDNFS-RLRNN configurations continues to increase as α is increased under the *Betw* caching policy. As expected, the NRR strategy incurs significantly greater overhead than the other two strategies. Across all the scenarios, Figure 6.9 shows that at $p = 0.3$, NDNFS-RLRNN reduces the network overhead by more than a factor of 4 compared to the NRR strategy. Furthermore, while NDNFS-RLRNN with low exploration incurs lower overhead compared to the ASF strategy in all the considered scenarios, for larger values of α, the network overhead of NDNFS-RLRNN at maximum exploration exceeds that of the ASF strategy.

Figure 6.10 shows the throughput achieved by the four forwarding strategies. We observe that the NDNFS-RLNN strategies can achieve a similar total number of successful deliveries as the

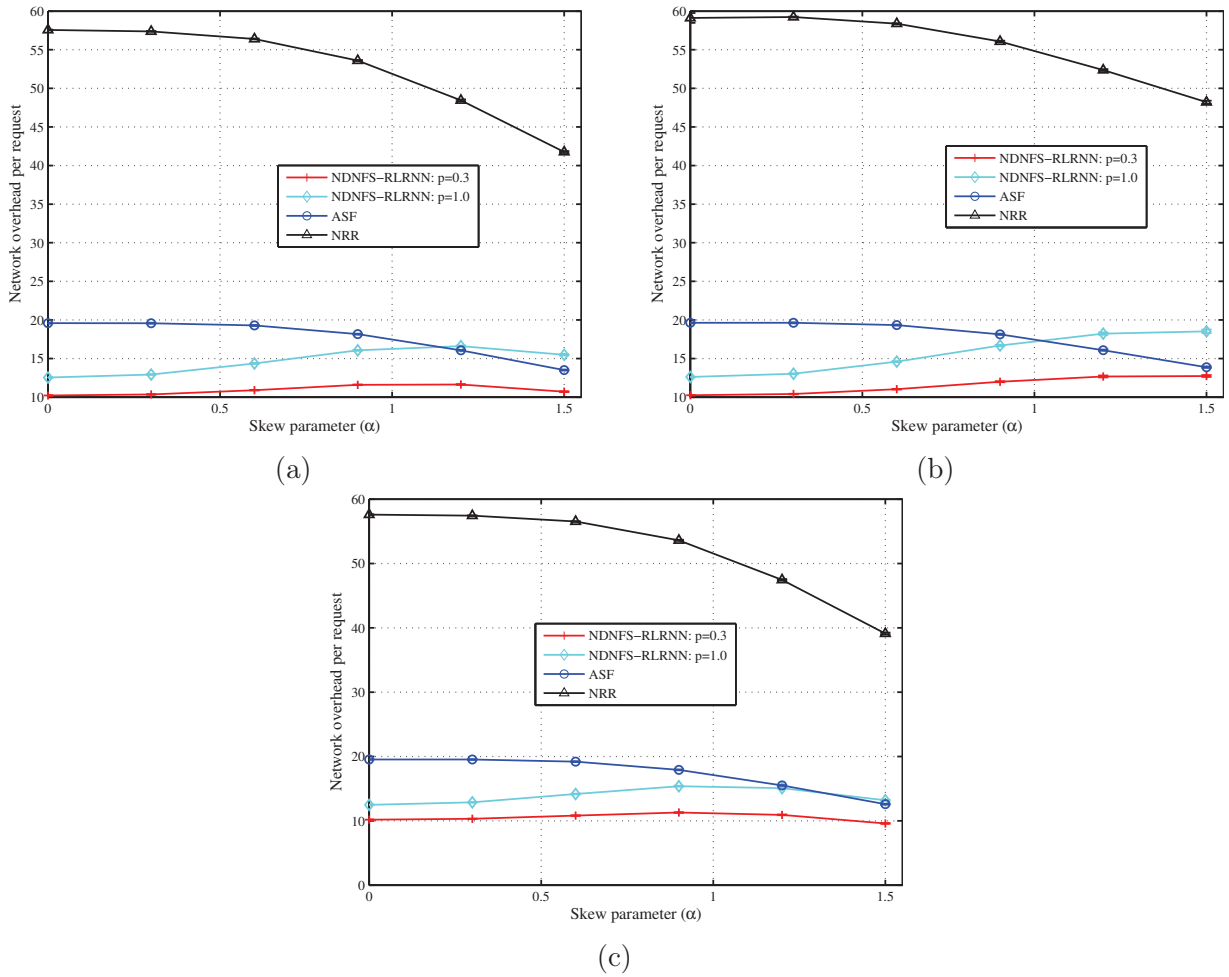


Figure 6.9: Network load performance of NDNFS-RLRNN ($p = 0.3$ and $p = 1.0$), ASF and NRR with uniform content popularity distributions, under: (**a**) LCE caching policy, (**b**) *Betw* caching policy and (**c**) *ProbCache* caching policy. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings: $q = 5.0, \alpha = 0.0, 0.3, 0.6, 0.9, 1.2$ and 1.5, and $N = 2000$. Each CR is equipped with a cache size of 0.4% of the content catalogue. The results reported are the average of 30 randomised simulation runs.

ASF strategy, which further supports the fact that our approach can match the routing layer's guarantees for content retrieval. We note that the throughput of the NRR strategy is lower than the other strategies, especially most noticeable under the *Betw* caching policy in Figure 6.10b. In the next section, we investigate and provide a likely explanation for this drop in throughput in the next section.

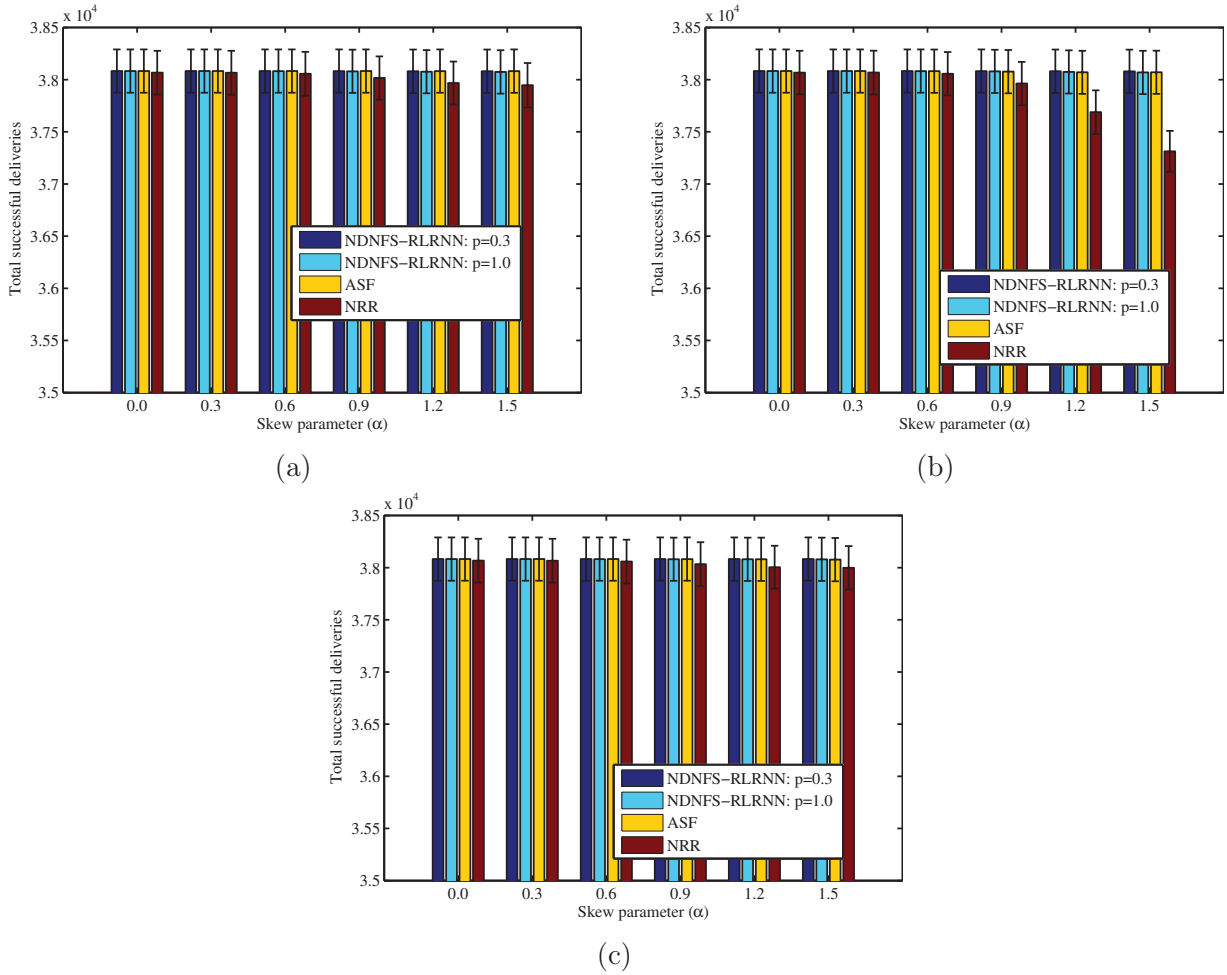
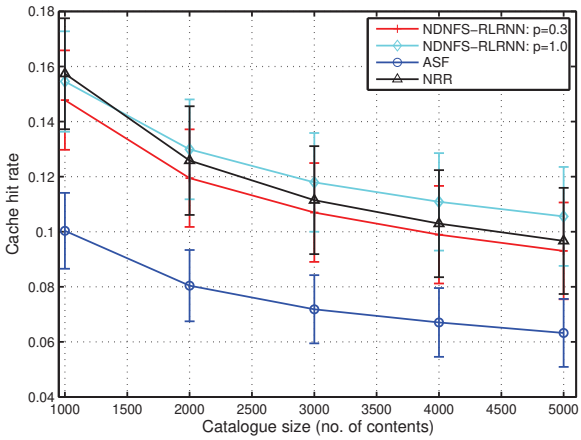


Figure 6.10: Throughput performance of NDNFS-RLRNN ($p = 0.3$ and $p = 1.0$), ASF and NRR with uniform content popularity distributions, under: (**a**) LCE caching policy, (**b**) *Betw* caching policy and (**c**) *ProbCache* caching policy. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings: $q = 5.0, \alpha = 0.0, 0.3, 0.6, 0.9, 1.2$ and 1.5, and $N = 2000$. Each CR is equipped with a cache size of 0.4% of the content catalogue. The results reported are the average of 30 randomised simulation runs.

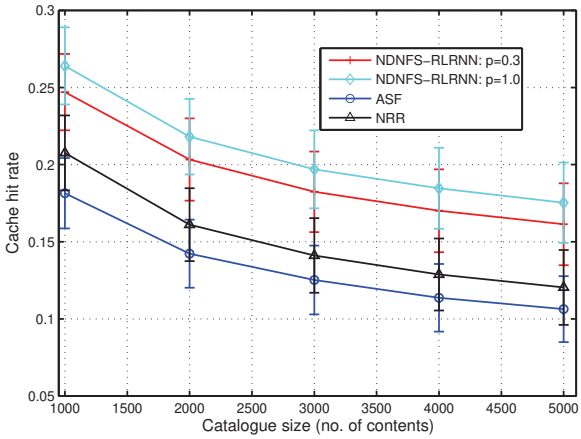
6.4.4 Performance evaluation for non-uniform request patterns

In Figures 6.11 - 6.13, we compare the NDNFS-RLRNN with both the ASF and NRR strategies under non-uniform request patterns. We achieve non-uniformity by randomly selecting the skew parameter for the MZipf distribution at each CR from the range $[0.6, 1.2]$; q is fixed at 5.0. We observe the impact of the cache size to catalogue size ratio on performance by fixing the cache size of each CR at 10 data objects while varing the catalogue size fom 1000 to 5000 data objects.

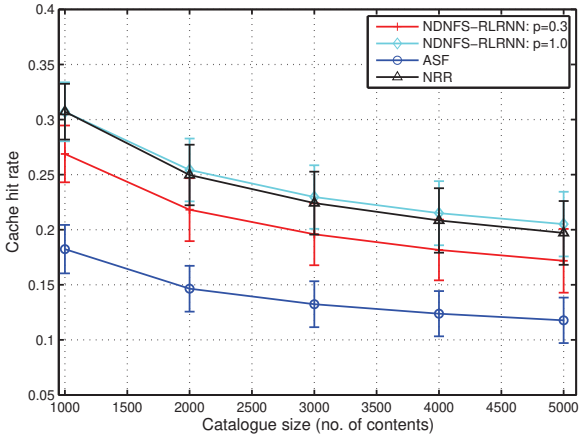
Figure 6.11 shows that even with low exploration, NDNFS-RLRNN can exploit in-network caching better than ASF. At $p = 0.3$, NDNFS-RLRNN achieves an improvement in the cache



(a)



(b)



(c)

Figure 6.11: Cache hit rate performance of NDNFS-RLRNN ($p = 0.3$ and $p = 1.0$), ASF and NRR for non-uniform request patterns, under: (**a**) LCE caching policy, (**b**) *Betw* caching policy and (**c**) *ProbCache* caching policy. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings with q fixed 5.0 and α selected randomly at each CR from the range $[0.6, 1.2]$. The cache size of each CR is fixed at 10 data objects while we vary the catalogue size fom 1000 to 5000 data objects. The results reported are the average of 30 randomised simulation runs.

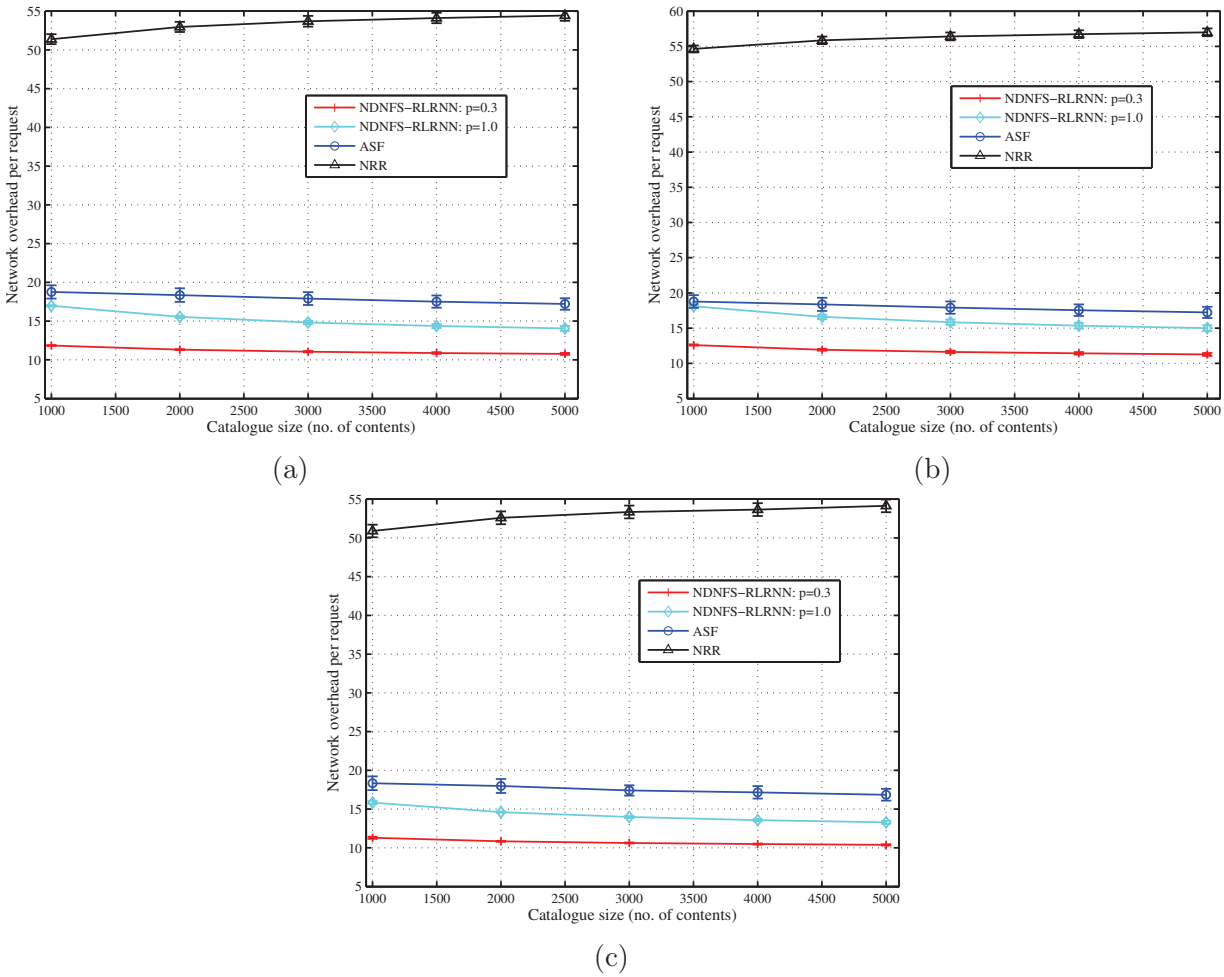


Figure 6.12: Network load performance of NDNFS-RLRNN ($p = 0.3$ and $p = 1.0$), ASF and NRR for non-uniform request patterns, under: (**a**) LCE caching policy, (**b**) *Betw* caching policy and (**c**) *ProbCache* caching policy. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings with q fixed 5.0 and α selected randomly at each CR from the range $[0.6, 1.2]$. The cache size of each CR is fixed at 10 data objects while we vary the catalogue size fom 1000 to 5000 data objects. The results reported are the average of 30 randomised simulation runs.

hit rate compared to ASF of between $47\% - 49\%$, $36\% - 52\%$ and $46\% - 49\%$ under the LCE policy, *Betw* policy and *ProbCache* policy, respectively. Actually, for the *Betw* policy, the percentage improvement increases as the catalogue size is increased. In comparison to the NRR strategy, low exploration in NDNFS-RLRNN only produces a better hit rate when the *Betw* policy is used. However, with the maximum exploration, NDNFS-RLRNN improves on the performance of the NRR strategy across all the considered caching strategies. Figure 6.11 shows that at $p = 1.0$, compared with NRR, NDNFS-RLRNN improves the cache hit rate by between $1\% - 2\%$, $27\% - 46\%$ and $1\% - 2\%$ under the LCE policy, *Betw* policy and *ProbCache* policy, respectively; while at $p = 0.3$, NDNFS-RLRNN improves the cache hit rate by between

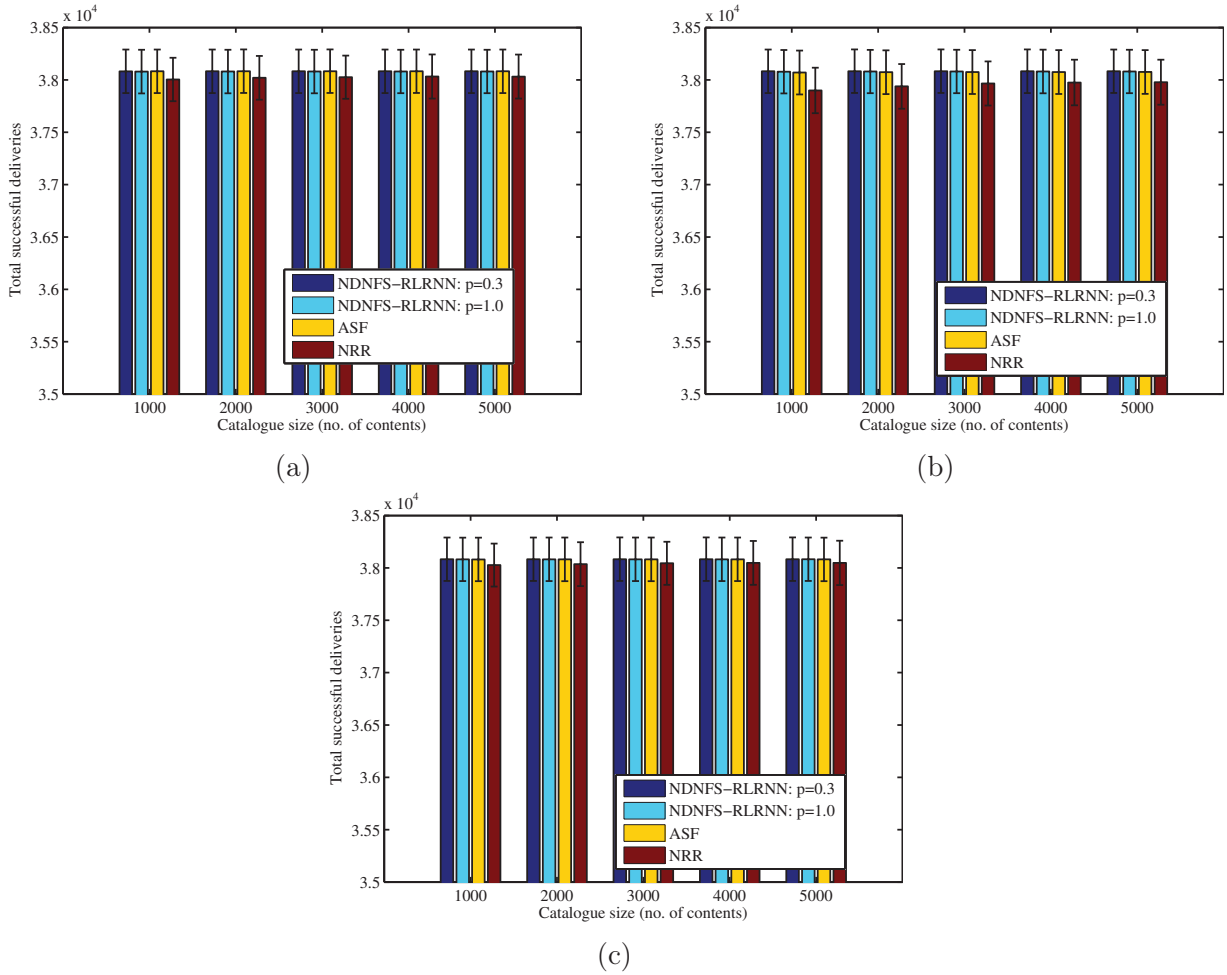


Figure 6.13: Throughput performance of NDNFS-RLRNN ($p = 0.3$ and $p = 1.0$), ASF and NRR for non-uniform request patterns, under: (**a**) LCE caching policy, (**b**) *Betw* caching policy and (**c**) *ProbCache* caching policy. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with the settings with q fixed 5.0 and α selected randomly at each CR from the range $[0.6, 1.2]$. The cache size of each CR is fixed at 10 data objects while we vary the catalogue size fom 1000 to 5000 data objects. The results reported are the average of 30 randomised simulation runs.

$18\% - 35\%$ under the *Betw* policy as shown in Figure 6.11b. Furthermore, we observe that for the *Betw* policy, the percentage improvements produced by the NDNFS-RLRNN algorithms over the NRR strategy increases as the catalogue size is increased. Overall, the cache hit performance of all the strategies diminishes as the cache size to catalogue size ration increases.

Figure 6.12 shows that even with maximum exploration, NDNFS-RLRNN incurs less overhead than the ASF strategy that probes every 3 seconds, under all the considered caching policies. Across all the caching policies, the NRR strategy's overhead is, on the average, about 4.8 and 3.6 times the overhead incured by NDNFS-RLRNN with low and maximum exploration,

respectively.

In Figure 6.13, we observe that the NDNFS-RLNN strategies again match the ASF strategy in the total number of successful deliveries. Interestingly, the NRR strategy does not achieve maximum throughput. In fact, in comparison with NDNFS-RLRNN, the total successful deliveries for NRR is less by an average of 60, 130, and 42 under the LCE policy, *Betw* policy and *ProbCache* policy, respectively. We believe this is caused by Interest aggregation, supporting the claims in [GLAMB15]. To further verify this hypothesis, we run simulations for the same scenario at $N = 1000$ and using the LCE policy, but with the NRR strategy without Interest aggregation. Table 6.1 compares the performance of the NRR strategy with and without aggregation.

Table 6.1: Effect of Interest aggregation on the NRR strategy.

	Cache hit rate	**Total successful deliveries**	**Network overhead**
NRR with Interest aggregation	0.1574	38004	51
NRR without Interest aggregation	0.0496	38080	96

We observe that without Interest aggregation, NRR delivers, on average, about 76 more Data packets than with Interest aggregation. Although, the overhead almost doubles and the cache hit rate is reduced by about a factor of 3. The significant deterioration in the cache hit rate performance can be explained by the cache pollution resulting from the combination of excessive flooding of requests and the LCE caching policy.

Non-uniform caching policy

In our final tests, we consider a flat network with non-uniform caching policies. In other words, there is no uniform caching policy across the network, and the caching policy at a CR is randomly chosen from the three that we have considered. This is a realistic scenario for an interconnection of networks. That is, each node in the topology actually represents a different domain under unique administration and rules.

Here, we consider the Geant topology with nodes and links. We also retain the same non-uniformity in the request pattern as in Section 6.4.4, but with the size of the catalogue fixed at $N = 2000$ contents. Furthermore, the network is served by 2 publishers or producers, and we randomise their locations in each simulation run. It is also assumed that each producer is able to satisfy all the requests sent to the CRs. Therefore, the paths installed in the FIB of each CR are the shortest paths to the producers.

Table 6.2 compares the average results from 30 simulation runs for the 3 forwarding strategies. For the NDNFSRL-RLRNN strategy, we use the full exploration configuration. Table 6.2 shows

Table 6.2: Performance comparison of NDNFS-RLRNN, ASF and NRR for non-uniform caching policy across the network.

Strategy	**Cache hit rate**	**Total successful deliveries**	**Network overhead**
NDNFS-RLRNN	0.1349	76226	12
NRR	0.1376	75991	124
ASF	0.0706	76228	15

that NDNFS-RLRNN when compared to the ASF strategy, improves the cache hit rate by 91% while matching ASF's throughput and using less packet overhead. Furthermore, NDNFSRL-RLRNN achieves a similar cache hit rate, higher throughput and reduces the overhead by a factor of 10 compared to the NRR strategy.

6.4.5 Summary of results

The results of our extensive simulations can be summarised as follows:

1 Firstly, the measures we introduced to address correctness significantly improved the throughput of the proposed NDNFS-RLRNN. The results also suggest that the NDN forwarding strategies must consider correctness issues if Interests are to be sent on randomly selected paths without any guarantees with respect to loop freedom. As a result, we did not consider similar approaches which use reinforcement learning to implement the NDN forwarding strategy module in our evaluations.

2 Secondly, the results also show that NDNFS-RLRNN exploits in-network caching significantly better than the ASF strategy which restricts its forwarding decisions to the routing layer preferences.

3 Thirdly, in comparison with the NRR forwarding strategy which tries to explore the entire network for every request, the results show that NDNFS-RLRNN with full exploration can match the cache hit rate performance while incurring significantly lower overhead. Interestingly, for the *Betw* caching policy where most of the caching is done at the most central nodes in a given network, NDNFS-RLRNN can search more intelligently that the NRR strategy and achieve better hit rates even at low exploration.

4 Finally, we must mention that the results also indicate that when the caching strategy is not effective, there is little or no benefit from using NDNFS-RLRNN. In other words, NDNFS-RLRNN could achieve better results with caching strategies that allow CRs to cooperate when making their caching decisions to reduce cache replication.

6.5 Conclusion

This chapter has addressed Information Centric Networks in the framework of Named Data Networking (NDN). We have proposed and evaluated an adaptive forwarding strategy, NDNFS-RLRNN for the NDN architecture which employs an online learning algorithm, reinforcement learning using the random neural network, to forward Interest packets. Our proposed approach is dynamic and does not persist on the links put forward by the routing protocol, so that it may better recognise the role of the forwarding plane to take advantage of the in-network caching capability of the NDN architecture.

In our tests, we compare NDNFS-RLRNN with two other forwarding strategies: one that restricts forwarding and probing of Interests to the interfaces suggested by the routing protocol and another that broadcasts each Interest. We also considered three different caching policies from the literature to evaluate the forwarding strategies under different caching behaviours.

We consider two ubiquitous caching policies exhibiting different cache eviction rate properties and a policy that bases its caching decisions on the network topological information.

Our results suggest that when in-network caching is effective, NDNFS-RLRNN strategy can achieve better delivery than a forwarding strategy that persists on the existing static routing layer preferences and a more efficient performance than a nearest replica forwarding strategy that floods requests. For a policy that caches contents at specific nodes based on the network topology, the results also show that even with a low probing frequency, NDNFS-RLRNN produces the best performance compared with the other forwarding strategies.

Chapter 7

Managing Crowds in Hazards with Dynamic Grouping

Emergency navigation algorithms for evacuees in confined spaces typically treat all evacuees in a homogeneous manner, using a common metric to select the best exit paths. In this chapter [1], we present a quality of service (QoS) driven routing algorithm, using CPN, to cater to the needs of different types of evacuees based on their age, mobility, and level of resistance to fatigue and hazard.

7.1 Introduction

High levels of occupancy and crowding in modern urbanised societies can aggravate destructive crowd behaviours during an emergency evacuation process and induce unnecessary fatalities and injuries. Hence, traditional emergency alarm systems which only alert civilians of emergencies are being superseded by emergency navigation systems which provide further guidance. Advancements in information technology (IT) have also contributed to the development of

[1] This chapter is as a result of joint work between Olumide Akinwande (author of this thesis) and Huibo Bi. Huibo was researching routing evacuees using different path metrics. Olumide proposed the dynamic grouping mechanism. The design of the path metrics and the simulations in the Chapter were jointly worked on by both Olumide and Huibo.

complex emergency management systems (EMS) [GGW12] based on in-situ wireless sensor networks (WSN).

Thus, substantial research has been conducted to understand and model the behaviour of crowds in both normal and emergency situations [VMC13, BMV04, SE12, YS07a, NSK+06, GSA11, BKSZ04, HFV00]. Accompanying this tendency, the design of distributed systems using sensor networks and computational resources to help direct people and crowds in emergency situations has been studied [GW12]. Also, there has been work done on cyber attacks that can take place in such circumstances [GL07]. The time needed to find exits or other specific objects which are hard to see or identify in a hazardous environment has also been studied using mathematical models [Gel10].

To optimise the design of crowded sites and evaluate the clearance time for all evacuees, various cellular automata and agent-based models have been employed to simulate grouping behaviours with respect to individuals' movement capabilities during emergency evacuations. In [SGS14], a heterogeneous cellular automata model mimics the evacuation process in a retirement house; evacuees initially belong to three groups (middle-aged people, nursing staff and older people), and groups are also formed dynamically due to the follow-the-leader effect. In [MWS14] grouping behaviours in evacuations are induced by introducing "bosons" into cells of the floor field cellular automaton [BKSZ04]; bosons are placed by evacuees as markers to increase the probability for other group members to reach some particular cell. The resulting simulations indicate that the evacuation time decreases with the increasing numbers of groups. More generally, individuals may need to be treated differently during an emergency: older people should choose the safest paths that will remain ahead of the spreading hazard, while agile individuals may be able to take advantage of the fastest paths, and may accept some element of risk. Research on robotic and autonomous systems has shown the advantages of cooperative behaviour among agents [GSSR97, BK04].

However, most of the evacuee routing algorithms consider a single criterion to choose paths for all evacuees without considering their specific requirements due to variance on age, mobility, level of resistance to hazard etc. The idea of treating evacuees differently during an emergency

in terms of the paths chosen for them was first explored in [BG14]. Thus, this chapter develops on this idea and investigates the use of dynamic grouping of evacuees based on their characteristics as a way to improve the outcome of an emergency evacuation. The chapter studies the improvements that could be offered by tailoring the evacuation strategy to diverse categories of evacuees and by treating evacuees in a distinct way based on their capabilities (e.g. mobility). To this effect, we will use the concepts of the Cognitive Packet Networks (CPN) which uses a neural algorithm-based technique for finding paths [GF99].

The remainder of the chapter is organised as follows. We first review the literature relevant to our work, followed by an outline of the dynamic grouping of evacuees in Section 7.3. Section 7.4 presents the CPN variations for the evacuee routing problem, and the routing metrics are presented in Section 7.4.1 and Section 7.4.1, respectively. The simulation models and assumptions are then described in Section 7.5, and the experimental results and discussion are presented in Section 7.6. Finally, we draw conclusions in Section 7.7.

7.2 Literature Review

The study of emergency evacuation in confined spaces, which was initially motivated by defence applications [GC98, GHK05] has attracted much attention owing to the potential of losses in terms of human lives and property during a disaster. Since previous research indicated that destructive crowd behaviours such as stampedes can lead to serious fatalities [HFV00], much work has been dedicated to investigating and designing crowd behaviour models [ZZL09, GGO$^+$99] based on cellular automata models [BMV04, YS07b], social force models [HM95], fluid-dynamics [Hen71, HFMV02] and agents [GJ05, Bon02]. Another tendency of this research field is emergency navigation, which concentrates on combining mathematical models [KPH05] or algorithms [GW13] with underlying sensing, communications and distributed real-time computation to guide evacuees to safety in a built environment. In situations with different types of individuals, due to different speeds and delays, some individuals may overtake and pass others [BGP84] leading to confusion in managing and accounting for the evacuees.

In this review, we mainly focus on emergency navigation, since our work relates to navigation algorithms in emergency situations.

Due to limitations in processing power, early emergency navigation systems are commonly computer-aided information reporting systems to assist emergency managers in making decisions [BKW84]. Associated emergency navigation algorithms at that time normally used purely mathematical models to simplify an evacuation process and seek optimal solutions. Thus in [CFS82] evacuation planning is considered as a minimum cost network flow problem that converts the original building graph to a time-expanded network. By solving the time-expanded network via a linear programming algorithm, evacuees can obtain optimal routes and achieve the shortest evacuation time.

With the fast development of information and communications technology (ICT), research then moved to the development of complex Emergency Cyber-Physical-Human systems to direct evacuees to exits with the aid of an on-site wireless sensor network (WSN). At the core of emergency navigation systems, various emergency navigation algorithms have been proposed such as network flow based algorithms [LGS05, LHS03], queueing model-based approaches [DG13, KS00], potential maintenance algorithms [GSSR97, BLA07, TPT06], biological-inspired approaches [GT08, JSFS09, LFLZ10] and prediction-based algorithms [HPB+10, RGS+15]. Network flow based algorithms commonly predict the upper bound of the evacuation time and convert the original building model to a time-expanded network by duplicating the original network for each discrete time unit. Then, linear programming or heuristic algorithms are used to compute the optimal evacuation plan. This approach can achieve the optimal solution but does not take the spreading of the hazard into consideration. By treating significant locations such as doorways or staircases as "servers", queueing model-based approaches [GM76], which generalise the Markovian models of computer systems [Gel73c], transfer building graphs to a queueing network to estimate congestion and evacuation delays. Potential-based algorithms normally can dynamically develop navigation paths by assigning attractive or repulsive potentials to the exits and hazards, and the evacuees move as a result of the net attraction-repulsion in various directions [LDRR03]. However, these approaches require constant information exchange to update navigation maps for evacuees, even when the maps are concentrated at a

few fixed nodes and shared with the evacuees to determine their paths. Biological-inspired approaches employ heuristics to search for routes [LFLZ10] such as genetic algorithms, where the "fitness" of a path is based on its length and the congestion or the hazards it may contain; initially shortest paths are selected based on distance, and then new paths may evolve incrementally through crossover and mutation. Prediction-based algorithms utilise Bayesian networks to anticipate the hazard or crowd dynamics in disasters [RGS+15] and infer the location of people and hazards. A novel e-infrastructure is presented in [HPB+10] to predict the spread of hazards based on predictive models and live sensory data in a faster-than-real-time manner.

Since many emergency response systems are based on wireless sensor networks (WSN), routing protocols have been borrowed or adapted from existing solutions. However, communications which are essential in this context can easily malfunction during emergencies, and in [GG12] a resilient emergency support system (ESS) is proposed to disseminate emergency messages among evacuees with the aid of opportunistic communications (Oppcomms). Experimental results indicate that this system is robust to network failures during an emergency. But because Oppcomms are susceptible to malicious attacks such as flooding or denial of service [GGW12], a defence mechanism that uses a combination of identity-based signatures (IBS) and content-based message verification to detect malicious nodes is proposed and an infrastructure-less emergency navigation system is presented in [GB14b] to guide evacuees with the aid of smart handsets and cloud servers. Also in [FG09] a WSN-based distributed emergency management system that uses Dijkstra's algorithm to calculate the shortest paths for evacuees is considered.

Sensor nodes (SNs) collect hazard information, while decision nodes (DNs) provide advice to evacuees through visual indicators or portable devices. To avoid a full graph search and reduce communication costs, in [Fil10] the system was modified by replacing Dijkstra's algorithm with the Cognitive Packet Network (CPN) [GXS99] routing algorithm.

7.3 Health-Aware Classification and Dynamic Grouping of Evacuees

Before outlining the CPN approach for evacuee routing, we describe the dynamic grouping mechanism. We classify evacuees into two groups: Class 1 and Class 2. Class 1 evacuees include those typically within the age range of 12 - 50 years. On the other hand, Class 2 evacuees those who are still individually mobile, but move more slowly such as children ($<$ 12 years), older individuals ($>$ 50 years), or who may have been weakened or hurt during the evacuation. The age range we have assumed for classifying the evacuees is informed simply by the widely accepted average ages of puberty and menopause.

To the best of our knowledge, previous studies in emergency navigation persist on a single decision algorithm during the whole evacuation process and do not adjust in accordance with individuals' physical conditions and their immediate environments. Therefore, in addition to tailoring the path selection metrics to each class of the evacuees, we also propose that evacuees switch groups during an evacuation. For instance, when the health level of an individual belonging to the Class 1 evacuees drops below a certain percentage of its original value, it should be moved to the safer Class 2 evacuees due to its reduced mobility and injury. The details are shown in Pseudocode 2 and a list of symbols used is summarised in Table 7.1.

Table 7.1: List of symbols used in the Pseudocode 2.

Notation	Definition
G_{id}	Represents the group ID of an evacuee
G_{one}	Represents the group ID of "Class 1" evacuees
G_{two}	Represents the group ID of "Class 2" evacuees
G_{con}	Represents the group ID of "congestion-ease" evacuees
H_o	Represents the initial health value of an evacuee
H_t	Represents the health value threshold that triggers the "Class-switching" event

The health value of an evacuee is affected by the fatigue level and exposure to the hazard. In reality, it can be calculated by a portable device carried by evacuees. The fatigue level is determined by the walking distance of an evacuee, which can be updated when reaching a sensor node. The impact of hazard can be evaluated by the hazard intensity of all the adjacent

Pseudocode 2 The process of changing an evacuee from "Class 1" to "Class 2". DN, decision node.

```
When an evacuee reaches the vicinity of a DN, obtain G_id of the evacuee
if G_id ∈ G_one then
    obtain the health value H_e, H_t of the evacuee
    if H_e < H_t then
        G_id ← G_two
```

sensors. Hence, the current health value of an evacuee can be obtained by using (7.1).

$$H_{c^+} = H_c - fD_w - h\frac{\sum_{i=1}^{n} H_i}{n} \tag{7.1}$$

where H_{c^+} represents the current health value of an evacuee and H_c represents the last health value, term f is a constant fatigue rate that coordinates the relationship between health value and walking distance D_w since last updates. Term n represents the number of adjacent sensors, and H_i is the associated hazard intensity. Term h is a constant that coordinates the relationship between the health value and the hazard intensity. In our simulations, we drop the fatigue term and assume $h = 1$ for Class 1 evacuees and $h = 1.25$ for Class 2 evacuees.

7.4 CPN for Evacuee Routing

The CPN architecture is modified to address the needs of emergency evacuation as follows. First, there are no DPs (dump packets) since the evacuees themselves are the "payload" that is being controlled by CPN. Two types of wireless nodes are used to sense (for the purpose of the SPs) and convey the information needed by CPN:

- Sensor Nodes (SNs) that sense the presence of hazards (e.g. fire, gas) and detect the presence of evacuees in their vicinity (e.g. via RFID or a smart tag that people may be carrying), are in communication with neighbouring DNs and provide them with the information that they have sense,
- Decision Nodes (DNs) that act as wireless CPN routers and transmitters for SPs (which search for evacuee paths) and ACKs (which bring back the sensed information) and pro-

vide advice to evacuees in their vicinity.

There will be a DN in each office or room in the building, and in a large room, there may be many DNs. Each DN functions as a CPN node and is placed in a fixed location (e.g. on the ceiling) known in advance to the software of the EMS. Thanks to wireless communication, each DN knows whether the DNs and SNs in its immediate environment are properly working, and this is part of the information that it uses to provide guidance to evacuees. Between any pair of DNs in a large room at least one SN is deployed, and there may be more deployed in the middle of two DNs (for instance at doors of rooms) to monitor the situation of the surrounding area.

Thanks to the neighbouring SNs, each DN knows the state of the link or hop to its neighbouring DNs, and it sends out SPs that move from DN to DN, to obtain the state of the paths to exits: thus each SP sent out by a DN will collect path information as it moves through DNs, while DNs themselves will know the state of each neighbouring hop segment from the SNs in their immediate vicinity. ACKs which are paired with specific SPs will head back from the exit destination to the nodes.

Thus using the CPN algorithm, the DN will select the best (i.e. the shortest among the safest, or overall safest) path from its own location to a safe exit. The CPN algorithm will also return an ACK packet from an exit to the DN that has sent out an SP when that SP reaches the exit with the path information (including path quality). Thus by sending out multiple SPs, each DN maintains a list of paths to exits together with the "age" of the path, and the path's quality metric. However, contrary to CPN, evacuees, will obtain advice successively from different DNs (by wireless or via direction signs) and will not use a fixed "source routed" path, and the evacuees' path will be updated as they move and receive new advice. Although all evacuees receive full path information, a movement depth value is set so that as long as there is no path blockage due to increased hazards or congestion, evacuees are encouraged to traverse a given number of nodes before using the newly obtained path update. The effect of the movement depth was well studied in [Fil10]. The authors showed, using extensive simulations, that where evacuees receive new path information at every DN, that is, movement depth is 1, path loops

are likely which can deteriorate overall performance. On the other hand, although a large movement depth reduces the possibilities, evacuees are more susceptible to using outdated path information. We adopt a movement depth of 2 which obtained the best trade-off performance in [Fil10].

Each SP is assigned a "maximum lifetime", which is simply the maximum number of hops that it is allowed to traverse before it is discarded. The purpose of the maximum lifetime is to reduce congestion by deleting "lost" SPs from the network. The maximum number of hops is set to the total number of DNs in the network plus one. Without any more information, this value is the smallest number of hops possible which guarantees that a loop has occurred.

7.4.1 Routing Metrics

The routing metrics that we define are the QoS goals used in the RNN based reinforcement learning algorithm of CPN. When an ACK brings back sensory data to the source node, the collected information will be used to compute the current values of the routing metrics, and the result will be used to update the weights of the RNN. We specifically use a time-oriented and a hazard oriented metric, as defined below.

In Figure 7.1, we show how the graph of the building is constructed with the nodes being the locations of the DNs, while the SNs, collectively called the set S, are placed on the edges between nodes. SNs provide real-time information regarding hazards. A DN i receives data from a set of nearby SNs, defined as the set N_i. A SN s belongs to N_i if the Euclidean distance between s and i is not greater than R: $N_i = \{s \in S : ||l(i) - l(s)|| \leq R\}$ where $l(.)$ denotes the "location" or cartesian coordinates of a DN or SN. Each SN s estimates the hazard intensity $H(s, \tau)$ at time τ of the edge where it is located:

$$H(s,\tau) = \begin{cases} 1 & \text{if no hazard is present} \\ k \cdot 10^3 & \text{otherwise} \end{cases}$$

where τ is the time at which the measurement is made, and k is an integer in the interval $[1, 8]$

that indicates the level or intensity of the hazard. For instance, this could be the temperature (when there is a fire), or the amount of gas that is detected. The 10^3 coefficient is chosen to ensure the effective length of an edge, which we define later in this section, is highly sensitive to the intensity of the hazard.

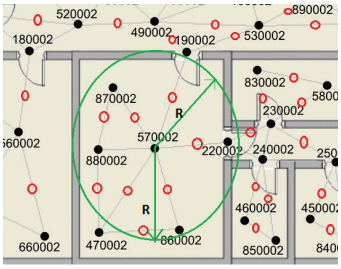


Figure 7.1: DNs are located on the black dots while SNs are positioned on the red rings. SNs in the green circle belong to N_{570002}.

Let $L_{(s)}$ be the physical length (say in meters) of the edge where a sensor s is located. Its effective length $L_e(s,\tau)$ at time τ will combine its real physical length with the hazards detected by s plus the average value the hazards in the vicinity defined by the radius R:

$$L_e(s,\tau) = L_{(s)}.[H(s,\tau) + \frac{\sum_{j:\ ||l(j)-l(s)||\leq R} H(j,\tau)}{|\{j \neq s :\ ||l(j)-l(s)|| \leq R\}|}] \tag{7.2}$$

If $R = 0$, the effective length becomes $L_e(s,\tau) = L(s).H(s,\tau)$.

A path exposed to fire (or another major hazard) can be labelled as such, in addition to the distance metric. However, in our case, the multiplicative factor used to signal the hazard is chosen so that hazardous paths will always have a distance greater than the safe paths in the built environment.

We now introduce two quality of service goals or metrics that the EMS will pursue to find the best paths for the evacuees. The Time Metric (TM) is quite simple and it seeks a fast evacuation path; it will be used by the EMS for the Class 1 evacuees who try to get out quickly but can afford to try a different path if they discover a hazard along their path. On the other hand, the Safety Metric (SM) will be used for the Class 2 or "weaker" evacuees who move more slowly and who are less able to try alternate routes if their initial routes turn out to be unsafe or clogged due to a hazard or congestion.

The Time Metric (TM)

The Time Metric (TM), denoted by $G(i, \pi, \tau)$, is used to choose egress paths that minimise the time it takes to evacuate the evacuees. A path π is a sequence of nodes and edges starting at some node i, so that we may write $\pi = (i_1, s_1, i_2, \ \ldots\ , s_n, i_{n+1})$ where $i_1 = i$ is the first node on the path, s_1 is the sensor on the edge from node i to the next node on the path, and so on until s_n, which is the sensor on the edge linking to the last node i_{n+1} on the path.

Each node can be viewed as a queue with a "server", where the service time is the time the evacuee needs to determine the next direction (by gaining suggestions from portable devices) plus the time it needs to physically move through the node. A recent study showed that Little's formula can be a useful approximation to estimate delays in emergency evacuations[DG13] even when transients are being considered. Assuming that this queue is stable (i.e. the arrival rate is smaller than the service rate), the average total time through a path can be estimated with Little's formula applied to each successive node in the path, including possible queueing times, and evaluated at time τ is then:

$$G_T(i, \pi, \tau) = \sum_{j=1}^{n} [\frac{L_{(s_j)}}{V} + \frac{q_{i_j}(\tau)}{a_{i_j}(\tau)}], \tag{7.3}$$

where V is the estimated speed of the evacuee, where the observed number of evacuees is q_{i_j} at node i_j (when this can be measured), and $a_{i_j}(\tau)$ is the observed arrival rate of evacuees at node i_j. Note that in many cases sensors will not be able to provide estimates of queue lengths and arrival rates, in which case these terms will be dropped.

The TM does not consider the spreading of the fire; it only seeks to guide evacuees to exits as soon as possible. However, the "virtual health" value, introduced in Section 7.3, which is roughly estimated by the evacuee's mobile device, helps evacuees that use the TM to adapt their strategy before they may enter a hazardous area.

The Safety Metric (SM)

The Safety Metric (SM) for path $\pi = (i_1, s_1, i_2, \ \ldots\ , s_n, i_{n+1})$ on the other hand, denoted by $G_S(i, \pi, \tau)$, is used to seek paths that help the evacuees avoid the hazards:

$$G_S(i, \pi, \tau) = \sum_{j=1}^{n} L_e(s_j, \tau) \tag{7.4}$$

Since the effective length of a path exposed to fire is always be greater than any other safe path in the building, the SM will help evacuees find the *shortest* among all the safe paths.

7.5 The Simulation Model and its Assumptions

To evaluate the proposed routing scheme for evacuees, we employ an existing Java-based distributed simulation tool, the Distributed Building Evacuation Simulator (DBES) [DFG10], and we use fire-related scenarios in the simulations. DBES can simulate large-scale environments (such as city neighbourhoods) [GG13] and is used to evaluate different courses of action in emergencies of varying danger and severity. As a multi-agent simulator, each entity in DBES is represented by a software agent that interacts with its environment. Figure 7.2 shows an example of the graphical user interface (GUI) of DBES with one "floor agent" in charge of managing the state of a given building's floor, and ten agents representing evacuees.

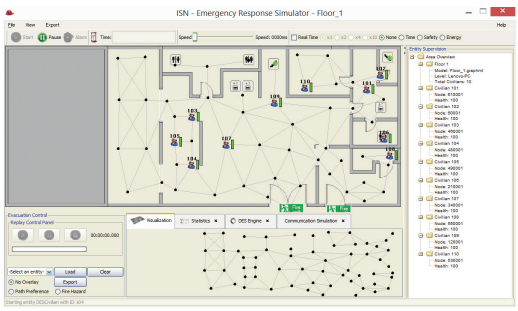


Figure 7.2: The GUI of the DBES.

7.5.1 Building Model

The building model in our experiments simulates the three lower floors of the EEE building at Imperial College London. The ground floor has a dimension of $24m$ by $45m$ while the other two floors have the same dimension of $24m$ by $60m$. The height between each floor is approximately $3m$. Figure 7.3 shows a graph representation of the building model. The second and third floors of the building being considered have more offices and rooms than the first floor which is essentially an exit area and a coffee shop; thus the second floor has 89 DNs or CPN nodes, the third floor has 92 DNs, while the first floor has just 59 DNs.

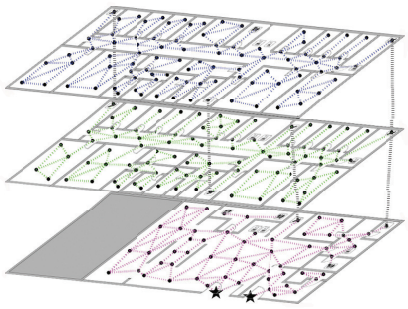


Figure 7.3: Graph representation of the building.

The vertices (black round dots) in Figure 7.3 represent locations where people can congregate such as rooms, doorways, and corridors while the two black stars on the first floor depict the exits. There is a total of 240 vertices on the graph, including the two exits, and at each vertex, we assume that a DN has been placed with SNs placed between each pair of DNs. The spaced horizontal lines linking the vertices are the graph edges representing the possible paths in the building, and on each edge there will be at least one SN. The edges connecting the two floors are the stairs in the building.

7.5.2 Modeling the Evacuees

The evacuees are assumed, for simplicity, to belong to two categories based on their age:

- The Class 1 agents represent evacuees typically within the age range of 12 - 50 years.

- The Class 2 agents represent evacuees who are still individually mobile, but move more slowly such as children, older individuals, or who may have been weakened or hurt during the evacuation.

The health level of each evacuee is initialised to a value of 100, and it is decreased over the course of the evacuation simulation based on fatigue and exposure to the hazard. Each category of evacuees is characterised by their speed and their resistance to fatigue and to the hazard. Table 7.2 illustrates the speeds of the two categories of evacuees.

The mobility model for Class 1 agents uses the empirical data found in the literature. For instance, the average human walking speed is $1.39m/s$ in general and is $1.19m/s$ in urban areas [Fru71]. We reduce this value to $1.05m/s$ because evacuees need to obtain suggestions from portable devices. The walking speed for the "upstair" direction is $0.51 \pm 0.10m/s$ in [Fra96] for a narrow staircase and $0.56 \pm 0.14m/s$ for a wider stair, while the down-stair motion speed is $0.72 \pm 0.29m/s$ for a narrow and $0.69 \pm 0.15m/s$ for a wider stair.

Table 7.2: Speeds of the two categories of evacuees.

Walking Speed	Class 1 Agent	Class 2 Agent
Direct	105 cm/s	84 cm/s
Upstairs	65 cm/s	53 cm/s
Downstairs	71 cm/s	57 cm/s

The speeds for the Class 2 agents are obtained by multiplying the corresponding Class 1 agent speed by a 0.8 factor. This factor is determined by the ratio between the walking speed of young adults and aged people [Tra97]. During the simulation, if the health level of an evacuee falls below 20%, its speeds drop to half of the values indicated in the table. We model the lower resistance of the Class 2 agents by multiplying the effect fatigue and hazard have on the health level of Class 1 agents by a factor of 1.5. The simulations are initialised by placing an evacuee in its initial location randomly at any of the nodes, and also each evacuee is initialised with probability 0.5 as belonging to either of the Classes. We also assume that all the evacuees are in possession of a wireless device that can receive path information from its neighbouring DNs.

7.5.3 Fire Source Location

The fire source location has a significant impact on the performance of the emergency navigation algorithms in the simulation. In the simulation (and perhaps also in a real situation), a fire that breaks out at a strategic location such as a staircase may result in all of the routing algorithms to operate equally poorly because of the potential for high congestion to create back pressure and further congestion on the higher floor(s).

To alleviate this issue by taking adequate precautions, we calculate the most "critical" nodes in the building by using the following definition: the *criticality rank* of a node, introduced in [DG13], is the number of shortest paths to the exit, starting from any node in the graph, that traverse the node. The highest ranked nodes by criticality for the graph used in our simulations are shown in Table 7.3.

Table 7.3: Most critical nodes in the building.

	Node Id	Count
1	410001	200
2	370001	103
3	360001	102
4	120001	101
5	20001	100

These top-ranked nodes form a path towards the exit that is located in the lobby on the first floor as shown in Figure 7.4. To evaluate the adaptiveness of the proposed algorithm, we choose Node 210001 as the fire source which is in an office on the first floor not far from the eastern staircase. By choosing this location, the fire will soon block this staircase, and this facilitates evaluating if the decision algorithms can adapt to the highly dynamic environment and discover the primary main channel marked out by the thick green lines in Figure 7.4.

7.6 Experiments and Results

We have carried out two experiments to investigate the potential improvements offered by routing two categories of evacuees using different metrics, varying health threshold H_t (which

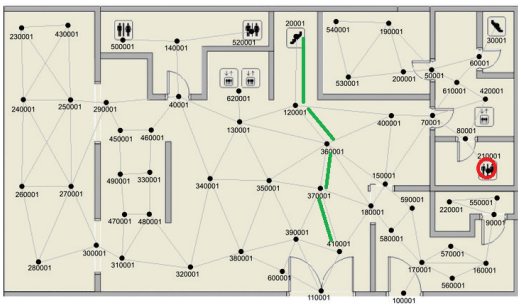


Figure 7.4: This shows the most critical nodes along the primary main channel marked out by the thick green lines and the fire node is indicated by the thick red ring.

is defined in Table 7.1) and diverse spatial information. The level of spatial information is determined by the operating communication range of the DNs, which is set by the variable R. In the first experiment, we set H_t to 50 to study the effect of customising metrics for different categories of evacuees under different levels of spatial information.

In the first two scenarios of this experiment, we use a single metric (time metric or safety metric with $R = 300$) in routing all the evacuees, while in the remaining four scenarios, we use one specific metric for each category of evacuees. In the second experiment, we set R to 300 to investigate the impact of using different health threshold H_t. For each scenario, we run 10 simulations with random distribution of the evacuees under four different levels of occupancy (30, 60, 90 and 120) in the aforementioned building model. Table 7.4 below gives a summary of the experiments that have been performed in the first experiment.

Table 7.4: Metrics used for class-based emergency evacuation.

Experiment 1	Evacuee type	Aim
CPN with safety metric (SM)	Class 1; Class 2	Safest path
CPN with time metric (TM)	Class 1; Class 2	Quickest path
CPN with safety and time metric (CM)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) with spatial information ($R = 300cm$)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) with spatial information ($R = 400cm$)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) with spatial information ($R = 500cm$)	Class 1; Class 2	Quickest path; safest path

We use the "average percentage of survivors" as the performance metric to evaluate the effectiveness of different algorithms and for each level of occupancy. An evacuee is considered to be

a "survivor" if it has a health level strictly greater than zero at the end of a simulation. The error bars in the plots represent minimum and maximum values.

7.6.1 Average percentage of survivors

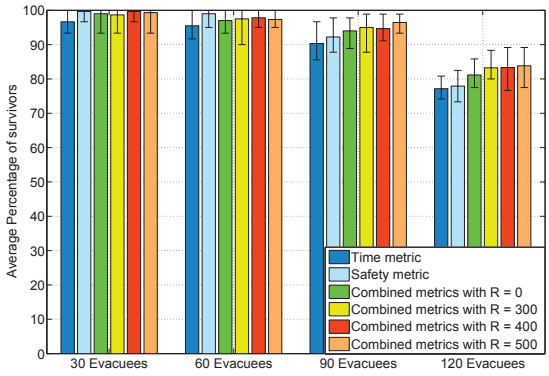


Figure 7.5: The average percentage of survivors for each scenario. The results are the average of 10 randomised simulation runs, and error bars show the min/max result in any of the 10 simulation runs.

Figure 7.5 shows the percentage of survivors in the first experiment, which employs time metric (TM), safety metric (SM) and combined metrics with diverse R (CM). TM gives the worst performance especially at low levels of occupancy (30 and 60 evacuees). This is because unlike the safety metric (SM), which tends to guide all the evacuees to the safest path, evacuees using TM may take the risk to traverse potential hazard areas to reduce the evacuation time. Hence, some evacuees may get injured or perish owing to the impact of the hazard. However, with the increase of occupancy rate, TM can reach the performance of SM because it can effectively ease congestion which occurs frequently in high population densities (90 and 120 evacuees). On the other hand, SM performs best at low occupancy rates because it is sensitive to the hazard and can choose safest paths for evacuees. However, the performance of SM degrades considerably in densely-populated environments because some paths with acceptable safety level are excluded. Hence, evacuees tend to congregate along several safest paths and generate high levels of congestion. In comparison with using one single metric, CM obtains overall best survival rates because it can tailor paths to evacuees with respect to their specific requirements. Furthermore, the concurrent use of two routing metrics can naturally distribute

evacuees and alleviate congestion. The results also indicate that CM with $R = 300$, 400 and 500 achieve better performance than CM with $R = 0$. This reflects that the use of spatial hazard information (R) has a positive impact on the performance of the algorithm. The reason is because the use of spatial information can generate a safe distance between evacuees and the spreading hazard.

7.6.2 The Effect of Dynamic Grouping

In the first experiment where we consider CM, a Class 1 evacuee whose health level falls below $H_t = 50$ will be immediately considered as a member of the second Class. To evaluate the effect of H_t, in the second experiment, as shown in Table 7.5, we concentrate on CM with $H_t = 0$, 30, 50, 70 and 90, respectively. When $H_t = 0$, two categories of evacuees will be guided with SM and TM separately and avoid changing of classes. This means that Class 1 evacuees will use TM throughout each simulation. Figure 7.6 shows the comparisons of average percentage of survivors with $R = 300$.

Table 7.5: The summary of experiments performed in the second experiment. Term R represents the level of spatial information and H_t denotes the health threshold for class-switching.

Experiment 2	Evacuee type	Aim
CPN with safety and time metric (CM) ($R = 300cm$, $H_t = 0$)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) ($R = 300cm$, $H_t = 30$)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) ($R = 300cm$, $H_t = 50$)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) ($R = 400cm$, $H_t = 70$)	Class 1; Class 2	Quickest path; safest path
CPN with safety and time metric (CM) ($R = 500cm$, $H_t = 90$)	Class 1; Class 2	Quickest path; safest path

These results indicate that the dynamic changing of classes generally has a positive impact on the system performance. In comparison with not changing classes ($H_t = 0$), dynamic grouping mechanism can achieve improved survival rates especially in high population densities. At low occupancy rates, CMs with different H_t achieve comparable results because when certain

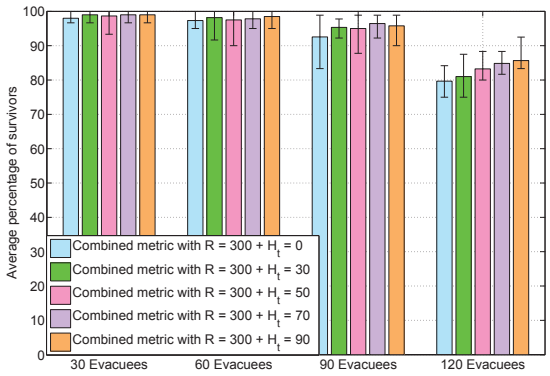


Figure 7.6: The average percentage of survivors for different H_t. The results are the average of 10 randomised simulation runs, and error bars show the min/max result in any of the 10 simulation runs.

distant evacuees are re-directed to a safe detour path, the spreading hazard may have blocked both staircases between floor 1 and floor 2. Hence, these evacuees have to traverse hazardous areas and suffer injuries and fatalities. On the other hand, at high occupancy rates, the survival rate increases with the increase of H_t. This is because CM with a larger H_t is more sensitive to the potential hazard and can direct evacuees away from hazardous zones earlier. On the contrary, if H_t is too small, evacuees may not switch classes in time and may suffer serious injury and reduced mobility before being re-routed.

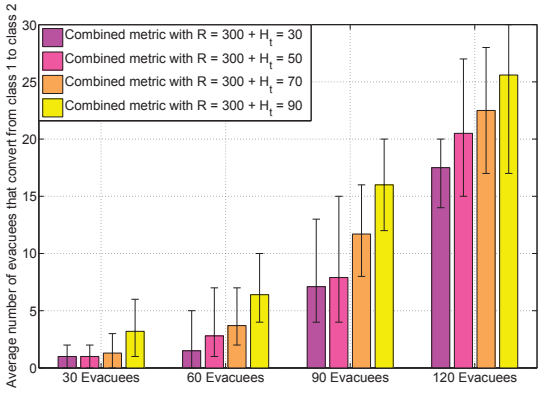


Figure 7.7: The average number of evacuees that convert from Class 1 to Class 2 during an evacuation process for each level of occupancy. The results are the average of 10 randomised simulation runs, and error bars show the min/max result in any of the 10 simulation runs.

Figure 7.7 shows the average number of evacuees that use dynamic grouping mechanism. It clearly shows that, with the increase of H_t, more evacuees use dynamic grouping mechanism and change from Class 1 to Class 2 during an evacuation process. Figure 7.8 presents the

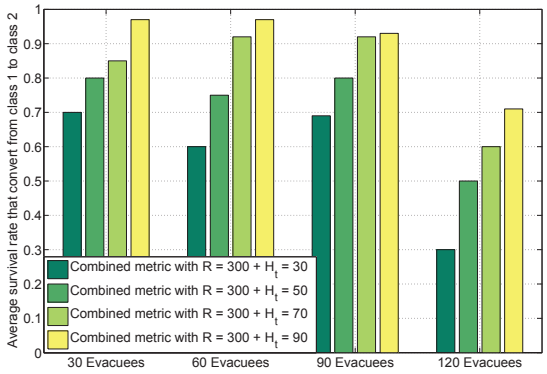


Figure 7.8: The average survival rate of Class-switching evacuees during an evacuation process for each level of occupancy. The results are the average of 10 randomised simulations.

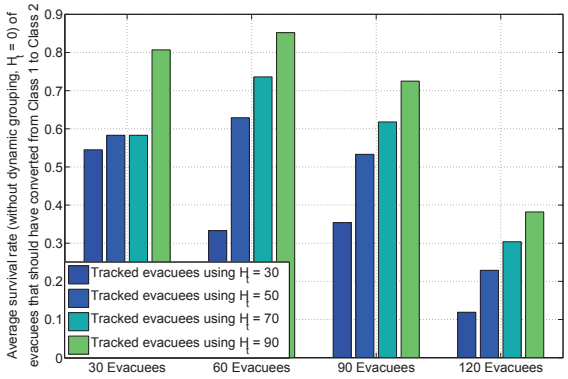


Figure 7.9: The average survival rate of tracked evacuees that should have converted from Class 1 to Class 2 in scenarios without dynamic grouping for the different values of H_t. For example, "tracked evacuees using $H_t = 30$" shows the survival rate of evacuees that should have changed Class when $H_t = 30$. The results are the average of 10 randomised simulation runs.

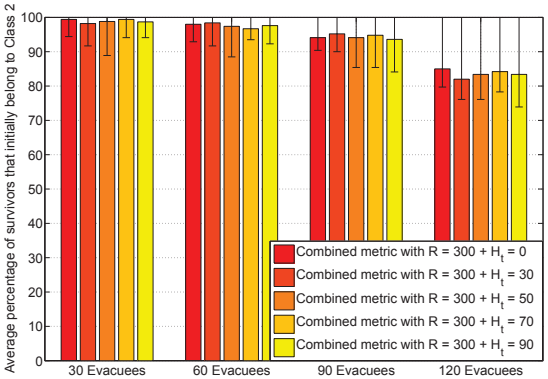


Figure 7.10: The average percentage of survivors of the original Class 2 evacuees for each level of occupancy. The results are the average of 10 randomised simulation runs, and error bars show the min/max result in any of the 10 simulation runs.

survival rate (S_c) between the number of class-switching survivors and the number of evacuees that employ dynamic grouping mechanism, which is defined in (7.5).

$$S_c = \frac{N_s}{N_c} \tag{7.5}$$

where N_c is the total number of Class 1 evacuees that convert to Class 2 during an evacuation process as shown in Figure 7.7. Term N_s represents the number of survivors that change from Class 1 to Class 2. As expected, the results indicate that the survival rates increase with the growth of H_t. Figure 7.7 and Figure 7.8 imply that the growth of H_t can increase the number of survivors that change from Class 1 to Class 2. This is because if H_t is small, evacuees may get injured and have no remaining time, mobility or possibility to change to a safe path.

Figure 7.9 shows the average survival rate of evacuees that should have converted from Class 1 to Class 2 in scenarios without dynamic grouping. By comparing with with Figure 7.8, we clearly see that the dynamic grouping mechanism can considerably improve the survival rate of these evacuees.

Figure 7.10 shows that the survival rates of the original Class 2 evacuees remain steady regardless of the variation of H_t. This indicates that the original Class 2 evacuees are not remarkably affected when more Class 1 evacuees join Class 2. In other words, although dynamic grouping mechanism converts a number of Class 1 evacuees to Class 2, the newly-assigned Class 2 evacuees do not influence the evacuation process of original Class 2 civilians.

In summary, the first experiment indicates that tailoring different QoS requirements to different classes of evacuees and dynamically assigning evacuees among classes with respect to the ongoing situation can improve the survival rates. Furthermore, the use of spatial information level R can improve the sensitivity of the safety metric and also increase the survival rate. In the second experiment, we investigate the effect of varying H_t. The results show that a properly selected H_t can significantly improve the survival rate of Class 1 evacuees. Meanwhile, H_t does not have an obvious impact on the original Class 2 evacuees. Furthermore, the average percentage of survivors for diverse H_t (shown in Figure 7.6) is not very obvious at low occupancy

rates because few evacuees will encounter the spreading hazard and switch their classes.

7.7 Conclusions

In this chapter, we propose an evacuation routing system based on CPN routing to direct different types of evacuees with respect to their ongoing health requirements. That is, an evacuee is allowed to change its class, and therefore during the evacuation process if it experiences a significant change in condition as a result of the hazard. The approach we propose is based on a situation where hazards, such as a fire, may move or change over time.

Our proposed system also combines spatial hazard information into the routing metrics used by CPN to try to prevent evacuees from being guided into hazards and to offer a better prediction of the spread and location of the hazard. Specifically, our approach groups the evacuees dynamically based both on their physical condition and the hazards in their surroundings.

The dynamic grouping mechanism is studied and evaluated via simulations. The simulation results indicate that this QoS-driven dynamic grouping system provides improved performance to achieve higher evacuee survival rates compared with treating all the evacuees in the same way, especially at high population densities. The simulation results also suggest that incorporating spatial hazard information can significantly improve the performance of the evacuation algorithm.

Chapter 8

Conclusion

8.1 Summary of Thesis Achievements

This thesis has studied the CPN architecture and explored novel applications for its routing algorithm to answer our research questions. Specifically, CPN routing is applied in ICN, a future Internet architecture proposal, and in an emergency navigation system for guiding evacuees in confined spaces.

ICN proposes a shift away from the current host-centric architecture of the Internet to one which treats data as fundamental. This thesis has studied the NDN architecture, one of the prominent ICN initiatives. NDN incorporates a stateful and intelligent forwarding plane in its architecture, in addition to a routing layer, for directing requests. The role of the forwarding plane is to leverage on both the routing layer and the observed delivery measurements to adaptively forward requests. Hence, the forwarding plane presents a useful framework for exploiting learning algorithms and adaptation in the network and a possible application of CPN routing. We, therefore, proposed and evaluated a forwarding strategy, using the Reinforcement learning with RNN algorithm inspired from CPN, for NDN-based networks that effectively searches for local cache hits. Our method leverages but does not restricts its search to the routing layer preferences. We also introduced measures in our proposed forwarding strategy to address possible correctness issues that could result from using paths that are not guaranteed

to be loop-free.

Furthermore, this thesis studied the content placement problem of ICN-based networks. Here, a network administrator must decide on the best contents to cache in his network and where to cache them, to best satisfy his users and minimise his costs. The problem is proposed and analysed as an on-demand problem such that the caching decisions directly follow the user requests. Approximate solutions using heuristics are proposed together with a simple cache management system for implementing these solutions.

Finally, this thesis studied an application of CPN routing in Emergency Cyber-Physical Systems. CPN's QoS capabilities are exploited to find suitable paths for different groups of evacuees using well-defined path metrics. The grouping of the evacuees is based on their ongoing health condition such that an evacuee can change his group, and, therefore, his routing metric if his health condition significantly changes during an evacuation. With the help of a WSN, hazard levels can be estimated, and the CPN paths can be communicated to the evacuees.

The following describes in more details the conclusions made through the thesis.

In Chapter 5, we studied a simple cache optimisation strategy, the on-demand cache management strategy (ODCMS), for an ICN-based network. ODCMS works on-demand such that caching decisions follow the user requests, enabling replacement decisions to be adapted in real-time and in a pragmatic way. Via simulations, ODCMS is evaluated and compared to simple caching strategies without any form of optimisation from the literature. Our results showed that ODCMS achieved significant improvement in the average network delay compared to the simple strategies. Furthermore, we also benchmarked ODCMS against a near-optimal solution that caches the most popular contents in the most central nodes in the network without replacement. In all the scenarios tested, ODCMS achieved average delay performances within 5% of the near-optimal performance.

Chapter 6 addressed adaptive request forwarding in NDN. We proposed the NDN forwarding strategy using Reinforcement learning with RNN (NDNFS-RLRNN) for adaptively forwarding Interest packets. NDNFS-RLRNN is dynamic and does not persist on the links put forward by

the routing protocol, so that it may better recognise the role of the forwarding plane to take advantage of the in-network caching capability of the NDN architecture. We also incorporate control measures in NDNFS-RLRNN to address correctness issues that may arise from using paths that do not guarantee loop freedom. Our initial results supported the earlier claim already put forward in the NDN literature that loops can go undetected. Because of the control measures we introduced, NDNFS-RLRNN can match the routing layer guarantees for content delivery. Furthermore, our results showed that when caching is effective, NDNFS-RLRNN can significantly hit more caches compared to a strategy that restricts its forwarding decisions to the routing preferences. NDNFS-RLRNN was able to match, with significantly lower overhead, the performance of a strategy that searches the entire network through flooding.

In Chapter 7, CPN routing was proposed for navigating different types of evacuees with respect to their ongoing health requirements during an emergency. Specifically, we considered two groups of evacuees: Class 1 evacuees comprising of young adults and Class 2 evacuees comprising children or older individuals or those might have been hurt during the evacuation. The routing system then tries to find the quickest paths for Class 1 evacuees and the safest paths for Class 2 evacuees using well-defined path metrics. In our simulations, we considered a spreading fire hazard and different levels of occupancy in the building. Our results showed that treating evacuees separately and using the dynamic grouping proposed can improve the survival chances of the evacuees.

8.2 Future Work

In this Section, we discuss the areas where our work can be extended and propose possible future research directions.

Chapter 5 addressed the problem of cache optimisation in information-centric networks. An on-demand cache management system is proposed that adapts the caching decisions in an ICN-based network in real-time and following the user requests. Our proposed approach uses a centralised system made up of cache management nodes to implement the cache optimisation.

Further work will focus on developing a more decentralised approach which will improve the scalability and robustness of the system. In addition, considering the effects of congestion and link failures is more realistic and will improve the evaluation of the system.

In Chapter 6, the work addressed adaptive Interest forwarding in the Named Data Networking (NDN) architecture. A Reinforcement learning algorithm using the Random Neural Network (RNN), inspired from CPN routing, is adapted to direct Interests and intelligently search local content stores for cache hits. In our evaluations, it was assumed that the NDN-based networks operated below congestion. Future work will focus on evaluating our approach for congested systems and in the presence of denial of service attacks. Such evaluations can also lead to improvements in the adaptive on-line policy and result in a better evaluation of the computational overhead of our adaptive on-line approach.

In Chapter 5 and Chapter 6 where we evaluated information-centric networks we made some simplifying assumptions including uniform content sizes and object-level caching. Randomising the sizes of the contents and considering chunk level caching could further enhance the fidelity of our results.

In Chapter 7, we presented a second application of CPN for implementing the routing system of an Emergency Cyber-Physical System for directing evacuees in confined spaces during an emergency. Specifically, the proposed system uses a quality of service (QoS) driven routing algorithm and tailors the evacuation strategy to diverse categories of evacuees. For further research, improving the accuracy of mobility models of the evacuees can increase the reality of the simulation model. For example, a model that considers the effect of density on the speed of the evacuees will improve the evaluation. In addition, energy consumption is a crucial factor in sensor networks. Therefore, factoring energy consumption into the path selection metrics and the lifetime of the sensor nodes can both improve the robustness of our system and enhance the results.

Bibliography

[AB15] Olumide J. Akinwande and Huibo Bi. Routing diverse crowds in emergency with dynamic grouping. In *2015 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops 2015, St. Louis, MO, USA, March 23-27, 2015*, pages 487–492, 2015.

[ABCdO96] V. Almeida, A. Bestavros, M. Crovella, and A. de Oliveira. Characterizing reference locality in the www. In *Fourth International Conference on Parallel and Distributed Information Systems*, pages 92–103, Dec 1996.

[ABG15] Olumide J. Akinwande, Huibo Bi, and Erol Gelenbe. Managing crowds in hazards with dynamic grouping. *IEEE Access*, 3:1060–1070, 2015.

[ADI+12] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, July 2012.

[ADU71] Alfred V. Aho, Peter J. Denning, and Jeffrey D. Ullman. Principles of optimal page replacement. *J. ACM*, 18(1):80–93, January 1971.

[AG18] Olumide J. Akinwande and Erol Gelenbe. A reinforcement learning approach to adaptive forwarding in named data networking. In *Computer and Information Sciences - 32nd International Symposium, ISCIS 2018, Held at the 24th IFIP World Computer Congress WCC 2018, Poznan, Poland, September 20-21, 2018, Proceedings*, pages 211–219, 2018.

[Aki18] Olumide J. Akinwande. Interest forwarding in named data networking using reinforcement learning. *Sensors*, 18(10):3354, 2018.

[AMM+13] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang. Interest flooding attack and countermeasures in named data networking. In *2013 IFIP Networking Conference*, pages 1–9, May 2013.

[ANO10] Somaya Arianfar, Pekka Nikander, and Jörg Ott. On content-centric router design and implications. In *Proceedings of the Re-Architecting the Internet Workshop*, ReARCH '10, pages 5:1–5:6, New York, NY, USA, 2010. ACM.

[ASZ+14] Alexander Afanasyev, Junxiao Shi, Beichuan Zhang, Lixia Zhang, Ilya Moiseenko, Yingdi Yu, Wentao Shang, Yi Huang, Jerald Paul Abraham, Steve DiBenedetto, et al. NFD developer's guide. *Dept. Comput. Sci., Univ. California, Los Angeles, Los Angeles, CA, USA, Tech. Rep. NDN-0021*, 2014.

[ASZ+18] Alexander Afanasyev, Junxiao Shi, Beichuan Zhang, Lixia Zhang, Ilya Moiseenko, Yingdi Yu, Wentao Shang, Yanbiao Li, Spyridon Mastorakis, Yi Huang, Jerald Paul Abraham, Steve DiBenedetto, et al. NFD developer's guide. Technical Report NDN-0021, Revision 8, NDN, July 2018.

[AVH07] J. Arkko, C. Vogt, and W. Haddad. Enhanced route optimization for mobile ipv6. RFC 4866, RFC Editor, May 2007.

[BA15] Huibo Bi and Olumide J. Akinwande. Multi-path routing in emergency with health-aware classification. In *2nd International Conference on Information and Communication Technologies for Disaster Management, ICT-DM 2015, Rennes, France, November 30 - December 2, 2015*, pages 109–115, 2015.

[BAG15] Huibo Bi, Olumide J. Akinwande, and Erol Gelenbe. Emergency navigation in confined spaces using dynamic grouping. In *9th International Conference on Next Generation Mobile Applications, Services and Technologies, NGMAST 2015, Cambridge, United Kingdom, September 9-11, 2015*, pages 120–125, 2015.

[BBC+98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated service, 1998.

[BCA+12] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu. A survey of naming and routing in information-centric networks. *IEEE Communications Magazine*, 50(12):44–53, December 2012.

[BCC+17] Robert Birke, Javier Cámara, Lydia Y. Chen, Lukas Esterle, Kurt Geihs, Erol Gelenbe, Holger Giese, Anders Robertsson, and Xiaoyun Zhu. Self-aware computing systems: Open challenges and future research directions. In *Self-Aware Computing Systems.*, pages 709–722. Springer, 2017.

[BCF+99] L. Breslau, Pei Cao, Li Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: evidence and implications. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 126–134 vol.1, Mar 1999.

[BCS94] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. RFC 1633 (Informational), June 1994.

[Bel66a] L. A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Systems Journal*, 5(2):78–101, 1966.

[Bel66b] L. A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Syst. J.*, 5(2):78–101, June 1966.

[BG92] Dimitri Bertsekas and Robert Gallager. *Data Networks (2Nd Ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1992.

[BG14] Huibo Bi and Erol Gelenbe. Routing diverse evacuees with the cognitive packet network algorithm. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on*, pages 291–296. IEEE, 2014.

[BGP84] François Baccelli, Erol Gelenbe, and Brigitte Plateau. An end-to-end approach to the resequencing problem. *J. ACM*, 31(3):474–485, June 1984.

[BGW10] S. Borst, V. Gupta, and A. Walid. Distributed caching algorithms for content distribution networks. In *2010 Proceedings IEEE INFOCOM*, pages 1–9, March 2010.

[BK04] Calin Belta and Vijay Kumar. Abstraction and control for groups of robots. *Robotics, IEEE Transactions on*, 20(5):865–875, 2004.

[BKST13] C. Barakat, A. Kalla, D. Saucez, and T. Turletti. Minimizing bandwidth on peering links with deflection in named data networking. In *2013 Third International Conference on Communications and Information Technology (ICCIT)*, pages 88–92, June 2013.

[BKSZ04] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A*, 295:507–525, 2004.

[BKW84] Salvatore Belardo, Kirk R Karwan, and William Wallace. An investigation of system design considerations for emergency management decision support. *Systems, Man and Cybernetics, IEEE Transactions on*, (6):795–804, 1984.

[BL93] Justin A. Boyan and Michael L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 671–678, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[BL94] Justin A Boyan and Michael L Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. *Advances in neural information processing systems*, pages 671–671, 1994.

[BL17] Abhishek Bhattacharjee and Daniel Lustig. *Architectural and Operating System Support for Virtual Memory*. Morgan & Claypool Publishers, 2017.

[BLA07] Matthew Barnes, Hugh Leather, and DK Arvind. Emergency evacuation using wireless sensor networks. In *Local Computer Networks, 2007. LCN 2007. 32nd IEEE Conference on*, pages 851–857. IEEE, 2007.

[BM16] I. V. Bastos and I. M. Moraes. A forwarding strategy based on reinforcement learning for content-centric networking. In *2016 7th International Conference on the Network of the Future (NOF)*, pages 1–5, Nov 2016.

[BMV04] S. Bandini, S. Manzoni, and G. Vizzari. Situated cellular agents: a model to simulate crowding dynamics. *IEICE Transactions on Information and Systems: Special Issue on Cellular Automata*, E87-D(3):669–676, 2004.

[Bon02] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 3):7280–7287, 2002.

[BSF15] C. Bernardini, T. Silverston, and O. Festor. A comparison of caching strategies for content centric networking. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2015.

[BWG16] O. Brun, L. Wang, and E. Gelenbe. Big data for autonomic intercontinental overlays. *IEEE Journal on Selected Areas in Communications*, 34(3):575–583, March 2016.

[CCGT13] A. Compagno, M. Conti, P. Gasti, and G. Tsudik. Poseidon: Mitigating interest flooding ddos attacks in named data networking. In *38th Annual IEEE Conference on Local Computer Networks*, pages 630–638, Oct 2013.

[CD73] Edward G. Coffman, Jr. and Peter J. Denning. *Operating Systems Theory*. Prentice Hall Professional Technical Reference, 1973.

[CDF+02] Bradley Cain, Dr. Steve E. Deering, Bill Fenner, Isidor Kouvelas, and Ajit Thyagarajan. Internet Group Management Protocol, Version 3. RFC 3376, October 2002.

[CFS82] LG Chalmet, RL Francis, and PB Saunders. Network models for building evacuation. *Fire Technology*, 18(1):90–113, 1982.

[CG00a] D.R. Cheriton and M. Gritter. Triad: A new next-generation internet architecture, 2000.

[CG00b] Christopher Cramer and Erol Gelenbe. Video quality and traffic qos in learning-based subsampled and receiver-interpolated video sequences. *IEEE Journal on Selected Areas in Communications*, 18(2):150–167, 2000.

[CGB96] C. Cramer, E. Gelenbe, and H. Bakircloglu. Low bit-rate video compression with neural networks and temporal subsampling. *Proceedings of the IEEE*, 84(10):1529–1543, Oct 1996.

[CGMP11] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino. Modeling data transfer in content-centric networking. In *2011 23rd International Teletraffic Congress (ITC)*, pages 111–118, Sept 2011.

[CGT13] Mauro Conti, Paolo Gasti, and Marco Teoli. A lightweight mechanism for detection of cache pollution attacks in named data networking. *Comput. Netw.*, 57(16):3178–3191, November 2013.

[CH03] Xi Cheng and Yi-Bin Hou. A study of genetic ant routing algorithm. In *Machine Learning and Cybernetics, 2003 International Conference on*, volume 4, pages 2041–2045. IEEE, 2003.

[Che14] Min Chen. Ndnc-ban: Supporting rich media healthcare services via named data networking in cloud-assisted wireless body area networks. *Information Sciences*, 284:142 – 156, 2014. Special issue on Cloud-assisted Wireless Body Area Networks.

[CHPP13] 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.

[Cis17] VNI Cisco. Cisco visual networking index: Forecast and methodology 2016–2021.(2017), 2017.

[CKR+07] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn, and Sue Moon. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 1–14, New York, NY, USA, 2007. ACM.

[CLP+12] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. 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, March 2012.

[CN98] Shigang Chen and K. Nahrstedt. An overview of quality of service routing for next-generation high-speed networks: problems and solutions. *IEEE Network*, 12(6):64–79, November 1998.

[Cos02] Andre Costa. *Analytic modelling of agent-based network routing algorithms*. PhD thesis, The University of Adelaide, 2002.

[CPC+13] Raffaele Chiocchetti, Diego Perino, Giovanna Carofiglio, Dario Rossi, and Giuseppe Rossini. Inform: A dynamic interest forwarding mechanism for information centric networking. In *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking*, ICN '13, pages 9–14, New York, NY, USA, 2013. ACM.

[CRW04] A. Carzaniga, M. J. Rutherford, and A. L. Wolf. A routing scheme for content-based networking. In *IEEE INFOCOM 2004*, volume 2, pages 918–928 vol.2, March 2004.

[CTW02] Hao Che, Ye Tung, and Zhijun Wang. Hierarchical web caching systems: modeling, design and experimental results. *IEEE Journal on Selected Areas in Communications*, 20(7):1305–1314, Sep 2002.

[CW02] Antonio Carzaniga and Alexander L. Wolf. Content-based networking: A new communication infrastructure. In Birgitta König-Ries, Kia Makki, Niki Pissinou, S.A.M. Makki, and Peter Scheuermann, editors, *Developing an Infrastructure for Mobile and Wireless Systems*, pages 59–68, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[CW03] Antonio Carzaniga and Alexander L. Wolf. Forwarding in a content-based network. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 163–174, New York, NY, USA, 2003. ACM.

[CW13] Abhishek Chanda and Cédric Westphal. Contentflow: Mapping content to flows in software defined networks. *CoRR*, abs/1302.1493, 2013.

[CY96] S Choi and Dit-Yan Yeung. Predictive q-routing: A memory-based reinforcement learning approach to adaptive tra c control. *Advances in Neural Information Processing Systems*, 8:945–951, 1996.

[CZM+17] Alberto Compagno, Xuan Zeng, Luca Muscariello, Giovanna Carofiglio, and Jordan Augé. Secure producer mobility in information-centric network. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ICN '17, pages 163–169, New York, NY, USA, 2017. ACM.

[DBGLA14] Ali Dabirmoghaddam, Maziar Mirzazad Barijough, and J.J. Garcia-Luna-Aceves. Understanding optimal caching and opportunistic caching at "the edge" of information-centric networks. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, ACM-ICN '14, pages 47–56, New York, NY, USA, 2014. ACM.

[DC+04] Gianni Di Caro et al. *Ant Colony Optimization and its application to adaptive routing in telecommunication networks*. PhD thesis, PhD thesis, Faculté des Sciences Appliquées, Université Libre de Bruxelles, Brussels, Belgium, 2004.

[DCD98] Gianni Di Caro and Marco Dorigo. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, pages 317–365, 1998.

[DDF+06] Simon Dobson, Spyros Denazis, Antonio Fernández, Dominique Gaïti, Erol Gelenbe, Fabio Massacci, Paddy Nixon, Fabrice Saffre, Nikita Schmidt, and Franco Zambonelli. A survey of autonomic communications. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 1(2):223–259, 2006.

[Den97] Peter J. Denning. Before memory was virtual. 1997.

[DFG10] Nikolaos Dimakis, Avgoustinos Filippoupolitis, and Erol Gelenbe. Distributed building evacuation simulator for smart emergency management. *The Computer Journal*, 53(9):1384–1400, 2010.

[DG13] Antoine Desmet and Erol Gelenbe. Graph and analytical models for emergency evacuation. *Future Internet*, 5(1):46–55, 2013.

[DG14] Antoine Desmet and Erol Gelenbe. A parametric study of cpn's convergence process. In Tadeusz Czachórski, Erol Gelenbe, and Ricardo Lent, editors, *Information Sciences and Systems 2014*, pages 13–20, Cham, 2014. Springer International Publishing.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[DT90] Asit Dan and Don Towsley. An approximate analysis of the lru and fifo buffer replacement schemes. *SIGMETRICS Perform. Eval. Rev.*, 18(1):143–152, April 1990.

[FCJS96] Ron Frederick, Stephen L. Casner, Van Jacobson, and Henning Schulzrinne. RTP: A Transport Protocol for Real-Time Applications. RFC 1889, January 1996.

[FG09] Avgoustinos Filippoupolitis and Erol Gelenbe. A distributed decision support system for building evacuation. In *Human System Interactions, 2009. HSI'09. 2nd Conference on*, pages 323–330. IEEE, 2009.

[FG16] Frederic Francois and Erol Gelenbe. Towards a cognitive routing engine for software defined networks. In *IEEE International Conference on Communications 2016*. IEEE, September 2016.

[FG17] Jean-Michel Fourneau and Erol Gelenbe. G-networks with adders. *Future Internet*, 9(3):34, 2017.

[FG18] Fréderic François and Erol Gelenbe. Optimizing secure SDN-enabled inter-data centre overlay networks through cognitive routing. In *MASCOTS 2016, IEEE Computer Society*, pages 283–288, 2018.

[Fil10] Avgoustinos Filippoupolitis. *Emergency Simulation and Decision Support Algorithms*. PhD thesis, Imperial College London (University of London), 2010.

[FLT+13] Seyed Kaveh Fayazbakhsh, Yin Lin, Amin Tootoonchian, Ali Ghodsi, Teemu Koponen, Bruce Maggs, K.C. Ng, Vyas Sekar, and Scott Shenker. Less pain, most of the gain: Incrementally deployable icn. *SIGCOMM Comput. Commun. Rev.*, 43(4):147–158, August 2013.

[FP7a] FP7 COMET project. Available: `http://www.comet-project.org/`.

[FP7b] FP7 CONVERGENCE project. Available: url=http://www.ict-convergence.eu/.

[FP7c] FP7 PURSUIT project. Available: url=http://www.fp7-pursuit.eu/PursuitWeb/.

[FP7d] FP7 SAIL project. Available: url=http://www.sail-project.eu/.

[Fra96] Håkan Frantzich. Study of movement on stairs during evacuation using video analysing techniques. *LUTVDG/TVBB–3079–SE*, 1996.

[Fru71] John J. Fruin. Pedestrian planning and design. Technical report, 1971.

[GB92] Erol GELENBE and Frederic BATTY. Minimum graph vertex covering with the random neural network. In OSMAN BALCI, RAMESH SHARDA, and STAVROS A. ZENIOS, editors, *Computer Science and Operations Research*, pages 139 – 147. Pergamon, Amsterdam, 1992.

[GB14a] Erol Gelenbe and Huibo Bi. Emergency navigation without an infrastructure. *Sensors*, 14(8):15142–15162, 2014.

[GB14b] Erol Gelenbe and Huibo Bi. Emergency navigation without an infrastructure. *Sensors*, 14(8):15142–15162, 2014.

[GB15] Peter Gusev and Jeff Burke. Ndn-rtc: Real-time videoconferencing over named data networking. In *Proceedings of the 2Nd ACM Conference on Information-Centric Networking*, ACM-ICN '15, pages 117–126, New York, NY, USA, 2015. ACM.

[GC98] Erol Gelenbe and Yonghuan Cao. Autonomous search for mines. *European Journal of Operational Research*, 108(2):319–333, 1998.

[GC01] Mark Gritter and David R. Cheriton. An architecture for content routing support in the internet. In *USITS'01: Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems*, pages 4–4, Berkeley, CA, USA, 2001. USENIX Association.

[Gel73a] E. Gelenbe. A unified approach to the evaluation of a class of replacement algorithms. *IEEE Transactions on Computers*, C-22(6):611–618, June 1973.

[Gel73b] Erol Gelenbe. The distribution of a program in primary and fast buffer storage. *Commun. ACM*, 16(7):431–434, July 1973.

[Gel73c] Erol Gelenbe. A unified approach to a class of page replacement algorithms. *IEEE Transactions on Computers*, 22(6):611–618, July 1973.

[Gel89] Erol Gelenbe. Random neural networks with negative and positive signals and product form solution. *Neural Comput.*, 1(4):502–510, December 1989.

[Gel90a] E. Gelenbe. Stability of the random neural network model. *Neural Computation*, 2(2):239–247, June 1990.

[Gel90b] Erol Gelenbe. Réseaux neuronaux aléatoires stables. *Comptes-rendus de l'Académie des Sciences. Série 2*, 310(3):177–180, 1990.

[Gel91] Erol Gelenbe. Product-form queueing networks with negative and positive customers. *Journal of Applied Probability*, 28(3):656–663, 1991.

[Gel93a] Erol Gelenbe. G-networks by triggered customer movement. *Journal of Applied Probability*, 30(3):742–748, 1993.

[Gel93b] Erol Gelenbe. G-networks with signals and batch removal. *Probability in the Engineering and Informational Sciences*, 7(3):335–342, 1993.

[Gel93c] Erol Gelenbe. Learning in the recurrent random neural network. *Neural Computation*, 5(1):154–164, 1993.

[Gel03] Erol Gelenbe. Sensible decisions based on qos. *Computational Management Science*, 1(1):1–14, Dec 2003.

[Gel04a] Erol Gelenbe. Cognitive routing in packet networks. In *International Conference on Neural Information Processing*, pages 625–632. Springer Berlin Heidelberg, 2004.

[Gel04b] S. Erol Gelenbe. Cognitive packet network. *US6804201*, (US6804201 B1 Application US 09/680,184), 2004.

[Gel09] Erol Gelenbe. Steps toward self-aware networks. *Commun. ACM*, 52(7):66–75, July 2009.

[Gel10] Erol Gelenbe. Search in unknown random environments. *Phys. Rev. E*, 82:061112, Dec 2010.

[Gel17] Erol Gelenbe. Self-aware networks: The cognitive packet network and its performance. In *Self-Aware Computing Systems*, pages 659–668. Springer, 2017.

[GF99] E. Gelenbe and J. M. Fourneau. Random neural networks with multiple classes of signals. *Neural Computation*, 11(4):953–963, May 1999.

[GFK96] Erol Gelenbe, Yutao Feng, and K Ranga R Krishnan. Neural network methods for volumetric magnetic resonance imaging of the human brain. *Proceedings of the IEEE*, 84(10):1488–1496, 1996.

[GG11] Gökçe Görbil and Erol Gelenbe. Opportunistic communications for emergency support systems. *Procedia Computer Science*, 5:39–47, 2011.

[GG12] Erol Gelenbe and Gokce Gorbil. Wireless networks in emergency management. In *Proceedings of the first ACM international workshop on Practical issues and applications in next generation wireless networks*, pages 1–6. ACM, 2012.

[GG13] Gokce Gorbil and Erol Gelenbe. Disruption tolerant communications for large scale emergency evacuation. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on*, pages 540–546. IEEE, 2013.

[GGL+04] Erol Gelenbe, Michael Gellman, Ricardo Lent, Peixiang Liu, and Pu Su. Autonomous smart routing for network qos. In *Autonomic Computing, 2004. Proceedings. International Conference on*, pages 232–239. IEEE, 2004.

[GGO+99] S Gwynne, ER Galea, M Owen, Peter J Lawrence, L Filippidis, et al. A review of the methodologies used in evacuation modelling. *Fire and Materials*, 23(6):383–388, 1999.

[GGS91] Erol Gelenbe, Peter Glynn, and Karl Sigman. Queues with negative arrivals. *Journal of Applied Probability*, 28(1):245–250, 1991.

[GGS97] E. Gelenbe, A. Ghanwani, and V. Srinivasan. Improved neural heuristics for multicast routing. *IEEE Journal on Selected Areas in Communications*, 15(2):147–155, Feb 1997.

[GGS03] E. Gelenbe, M. Gellman, and Pu Su. Self-awareness and adaptivity for quality of service. In *Proceedings of the Eighth IEEE Symposium on Computers and Communications. ISCC 2003*, pages 3–9 vol.1, July 2003.

[GGW12] E. Gelenbe, G. Gorbil, and F. Wu. Emergency cyber-physical-human systems. In *2012 21st International Conference on Computer Communications and Networks (ICCCN)*, pages 1–7, July 2012.

[GH02] Erol Gelenbe and Khaled F Hussain. Learning in the multiple class random neural network. *IEEE Transactions on Neural Networks*, 13(6):1257–1267, 2002.

[GHK05] Erol Gelenbe, Khaled Hussain, and Varol Kaptan. Simulating autonomous agents in augmented reality. *Journal of Systems and Software*, 74(3):255–268, 2005.

[GhMdL14] Erol Gelenbe, Zhi hong Mao, and Yan da Li. Function approximation by random neural networks with a bounded number of layers. *Differential Equations and Dynamical Systems*, 12(1):143–170, Jan 2014.

[GJ05] Robert L Goldstone and Marco A Janssen. Computational models of collective behavior. *Trends in cognitive sciences*, 9(9):424–430, 2005.

[GK14] Erol Gelenbe and Zarina Kazhmaganbetova. Cognitive packet network for bilateral asymmetric connections. *IEEE Transactions on Industrial Informatics*, 10(3):1717–1725, 2014.

[GKP93] E. Gelenbe, V. Koubi, and F. Pekergin. Dynamical random neural network approach to the traveling salesman problem. In *Proceedings of IEEE Systems Man and Cybernetics Conference - SMC*, volume 2, pages 630–635 vol.2, Oct 1993.

[GL04] Erol Gelenbe and Ricardo Lent. Power-aware ad hoc cognitive packet networks. *Ad Hoc Networks*, 2(3):205–216, 2004.

[GL05] E. Gelenbe and Peixiang Liu. Qos and routing in the cognitive packet network. In *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*, pages 517–521, June 2005.

[GL07] Erol Gelenbe and George Loukas. A self-aware approach to denial of service defence. *Computer Networks*, 51(5):1299–1314, 2007.

[GLAMB15] J.J. Garcia-Luna-Aceves and Maziar Mirzazad-Barijough. Enabling correct interest forwarding and retransmissions in a content centric network. In *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, ANCS '15, pages 135–146, Washington, DC, USA, 2015. IEEE Computer Society.

[GLL06] Erol Gelenbe, Peixiang Liu, and Jeremy LainLaine. Genetic algorithms for route discovery. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(6):1247–1254, 2006.

[GLMX02] Erol Gelenbe, Ricardo Lent, Alfonso Montuori, and Zhiguang Xu. Cognitive packet networks: Qos and performance. In *Modeling, Analysis and Simulation of Computer and Telecommunications Systems, 2002. MASCOTS 2002. Proceedings. 10th IEEE International Symposium on*, pages 3–9. IEEE, 2002.

[GLN04] Erol Gelenbe, Ricardo Lent, and Arturo Nunez. Self-aware networks and qos. *Proceedings of the IEEE*, 92(9):1478–1489, Aug 2004.

[GLX01a] Erol Gelenbe, Ricardo Lent, and Zhiguang Xu. Design and performance of cognitive packet networks. *Performance Evaluation*, 46(2):155–176, 2001.

[GLX01b] Erol Gelenbe, Ricardo Lent, and Zhiguang Xu. Measurement and performance of a cognitive packet network. *Computer Networks*, 37(6):691–701, 2001.

[GM76] Erol Gelenbe and Richard R. Muntz. Probabilistic models of computer systems, part i (exact results). *Acta Informatica*, 7:35–60, 1976.

[GM10] Erol Gelenbe and Isi Mitrani. *Analysis and Synthesis of Computer Systems: Texts)*. Imperial College Press, London, UK, UK, 2nd edition, 2010.

[GM11] Erol Gelenbe and Toktam Mahmoodi. Energy-aware routing in the cognitive packet network. *Energy*, pages 7–11, May 2011.

[GML99] Erol Gelenbe, Zhi-Hong Mao, and Yan-Da Li. Function approximation with spiked random networks. *IEEE Transactions on Neural Networks*, 10(1):3–9, 1999.

[GN08] E. Gelenbe and E. C. . Ngai. Adaptive qos routing for significant events in wireless sensor networks. In *2008 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, pages 410–415, Sept 2008.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[GP98] Erol Gelenbe and Guy Pujolle. *Introduction to networks of queues*. John Wiley Ltd., 1998.

[GPP+14] G. Grassi, D. Pesavento, G. Pau, R. Vuyyuru, R. Wakikawa, and L. Zhang. Vanet via named data networking. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 410–415, April 2014.

[GS92] Erol Gelenbe and Rolf Schassberger. Stability of product form g-networks. *Probability in the Engineering and Informational Sciences*, 6(3):271–276, 1992.

[GSA11] I. Georgoudas, G. Sirakoulis, and I. Andreadis. An anticipative crowd management system preventing clogging in exits during pedestrian evacuation process. *IEEE Systems Journal*, 5(1):129–141, 2011.

[GSCG96] Erol Gelenbe, Mert Sungur, Christopher Cramer, and Pamir Gelenbe. Traffic and video quality with adaptive neural compression. *Multimedia Systems*, 4(6):357–369, 1996.

[GSSR97] Erol Gelenbe, Nestor Schmajuk, John Staddon, and John Reif. Autonomous search by robots and animals: A survey. *Robotics and Autonomous Systems*, 22(1):23–34, 1997.

[GSX01] Erol Gelenbe, Esin Seref, and Zhiguang Xu. Simulation with learning agents. *Proceedings of the IEEE*, 89(2):148–157, 2001.

[GT08] Erol Gelenbe and Stelios Timotheou. Random neural networks with synchronized interactions. *Neural Computation*, 20(9):2308–2324, 2008. PMID: 18386985.

[GW12] Erol Gelenbe and Fang-Jing Wu. Large scale simulation for human evacuation and rescue. *Computers & Mathematics with Applications*, 64(12):3869–3880, 2012.

[GW13] Erol Gelenbe and Fang-Jing Wu. Future research on cyber-physical emergency management systems. *Future Internet*, 5(3):336–354, 2013.

[GW16] Erol Gelenbe and Lan Wang. TAP: A task allocation platform for the eu fp7 panacea project. In *Advances in Service-Oriented and Cloud Computing: Workshops of ESOCC 2015, Taormina, Italy, September 15-17, 2015, Revised Selected Papers*, volume 567, page 425. Springer, 2016.

[GXS99] Erol Gelenbe, Zhiguang Xu, and Esin Seref. Cognitive packet networks. In *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on*, pages 47–54. IEEE, 1999.

[GXS12] Shuo Guo, Haiyong Xie, and Guangyu Shi. Collaborative forwarding and caching in content centric networks. In Robert Bestak, Lukas Kencl, Li Erran Li, Joerg Widmer, and Hao Yin, editors, *NETWORKING 2012*, pages 41–55, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[GYCG18] Ingrid Grenet, Yonghua Yin, Jean-Paul Comet, and Erol Gelenbe. Machine learning to predict toxicity of compounds. In *27th Annual International Conference on Artificial Neural Networks, ICANN18, accepted for publication*. Springer Verlang, 2018.

[HAA+13] A K M Mahmudul Hoque, Syed Obaid Amin, Adam Alyyan, Beichuan Zhang, Lixia Zhang, and Lan Wang. Nlsr: Named-data link state routing protocol. In *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking*, ICN '13, pages 15–20, New York, NY, USA, 2013. ACM.

[Hal00] Ugur Halici. Reinforcement learning with internal expectation for the random neural network. *European Journal of Operational Research*, 126(2):288 – 307, 2000.

[HB91] C Hedrick and L Bosack. An introduction to IGRP. *Rutgers-The State University of New Jersey Technical Publication, Laboratory for Computer Science*, 1991.

[Hed88] C. L. Hedrick. Routing information protocol. 1988.

[Hen71] LF Henderson. The statistics of crowd fluids. *Nature*, 229:381–383, 1971.

[Hey08] L. A. Hey. Power aware smart routing in wireless sensor networks. In *2008 Next Generation Internet Networks*, pages 195–202, April 2008.

[HFMV02] Dirk Helbing, Illes J Farkas, Peter Molnar, and Tamás Vicsek. Simulation of pedestrian crowds in normal and evacuation situations. *Pedestrian and evacuation dynamics*, 21:21–58, 2002.

[HFV00] D. Helbing, I. Farkas, and T. Vicsek. Simulating dynamical features of escape panic. *Nature*, 407:487–490, 2000.

[HG09] L. Hey and E. Gelenbe. Adaptive packet prioritisation for wireless sensor networks. In *2009 Next Generation Internet Networks*, pages 1–7, July 2009.

[HGLA15] Ehsan Hemmati and J.J. Garcia-Luna-Aceves. A new approach to name-based link-state routing for information-centric networks. In *Proceedings of the 2Nd ACM Conference on Information-Centric Networking*, ACM-ICN '15, pages 29–38, New York, NY, USA, 2015. ACM.

[HKL+06] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J.A. Stankovic, T.F. Abdelzaher, J. Hui, and B. Krogh. Vigilnet: An integrated sensor network system for energy-efficient surveillance. *ACM Transactions on Sensor Networks*, 2(1):1–38, 2006. cited By 307.

[HM95] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[HPB+10] Liangxiu Han, Stephen Potter, George Beckett, Gavin Pringle, Stephen Welch, Sung-Han Koo, Gerhard Wickler, Asif Usmani, José L Torero, and Austin Tate. Firegrid: an e-infrastructure for next-generation emergency response support. *Journal of Parallel and Distributed Computing*, 70(11):1128–1141, 2010.

[HS08] M. Hefeeda and O. Saleh. Traffic modeling and proportional partial caching for peer-to-peer systems. *IEEE/ACM Transactions on Networking*, 16(6):1447–1460, Dec 2008.

[HSGK98] Martin Heusse, Dominique Snyers, Sylvain Guerin, and Pascale Kuntz. Adaptive agent-driven routing and load balancing in communication networks. *Advances in Complex Systems*, 1(02n03):237–254, 1998.

[IY17] Stratis Ioannidis and Edmund Yeh. Jointly optimal routing and caching for arbitrary network topologies. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ICN '17, pages 77–87, New York, NY, USA, 2017. ACM.

[JB00] S. Jin and A. Bestavros. Sources and characteristics of web temporal locality. In *Proceedings 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (Cat. No.PR00728)*, pages 28–35, 2000.

[JSFS09] A. Jankowska, M. C. Schut, and N. Ferreira-Schut. A wireless actuator-sensor neural network for evacuation routing. In *Sensor Technologies and Applications, 2009. SENSORCOMM'09. Third International Conference on*, pages 139–144. IEEE, 2009.

[JST+09] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '09, pages 1–12, New York, NY, USA, 2009. ACM.

[Kan15] B. Kantarci. Cyber-physical alternate route recommendation system for paramedics in an urban area. In *2015 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2155–2160, March 2015.

[KCC+07] 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*, SIGCOMM '07, pages 181–192, New York, NY, USA, 2007. ACM.

[KM97] Shailesh Kumar and Risto Miikkulainen. Dual reinforcement q-routing: An on-line adaptive routing algorithm. In *Artificial neural networks in engineering*, 1997.

[KNF+11] S. Knight, H.X. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *Selected Areas in Communications, IEEE Journal on*, 29(9):1765 –1775, october 2011.

[KPH05] Erica D Kuligowski, Richard D Peacock, and BL Hoskins. *A review of building evacuation models*. US Department of Commerce, National Institute of Standards and Technology, 2005.

[KRR02] Jussi Kangasharju, James Roberts, and Keith W. Ross. Object replication strategies in content distribution networks. *Comput. Commun.*, 25(4):376–383, March 2002.

[KS00] Laoucine Kerbache and J MacGregor Smith. Multi-objective routing within large scale facilities using open finite queueing networks. *European Journal of Operational Research*, 121(1):105–123, 2000.

[KXP11] Konstantinos Katsaros, George Xylomenos, and George C. Polyzos. Multicache: An overlay architecture for information-centric networking. *Comput. Netw.*, 55(4):936–947, March 2011.

[KY05] Sara Khodayari and Mohammad Javad Yazdanpanah. Network routing based on reinforcement learning in dynamically changing networks. In *Tools with Artificial*

Intelligence, 2005. ICTAI 05. 17th IEEE International Conference on, pages 5–pp. IEEE, 2005.

[KZ89] Atul Khanna and John Zinky. The revised ARPANET routing metric. *ACM SIGCOMM Computer Communication Review*, 19(4):45–56, 1989.

[LCLZ15] Tao Li, Jiannong Cao, Junbin Liang, and Junhao Zheng. Towards context-aware medical cyber-physical systems: design methodology and a case study. *Cyber-Physical Systems*, 1(1):5–23, 2015.

[LDRR03] Qun Li, Michael De Rosa, and Daniela Rus. Distributed algorithms for guiding navigation across a sensor network. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 313–325. ACM, 2003.

[Len03] Ricardo Lent. *On The Design and Performance of Cognitive Packets over Wired Networks and Mobile Ad Hoc Networks*. PhD thesis, University of Central Florida, 2003.

[LFLZ10] Qiuping Li, Zhixiang Fang, Qingquan Li, and Xinlu Zong. Multiobjective evacuation route assignment model based on genetic algorithm. In *Geoinformatics, 2010 18th International Conference on*, pages 1–5. IEEE, 2010.

[LG07] P. Liu and E. Gelenbe. Recursive routing in the cognitive packet network. In *2007 3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities*, pages 1–6, May 2007.

[LGS05] Qingsong Lu, Betsy George, and Shashi Shekhar. Capacity constrained routing algorithms for evacuation planning: A summary of results. In *Advances in spatial and temporal databases*, pages 291–307. Springer, 2005.

[LGZ+16] V. Lehman, A. Gawande, B. Zhang, L. Zhang, R. Aldecoa, D. Krioukov, and L. Wang. An experimental investigation of hyperbolic routing with a smart forwarding plane in ndn. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, pages 1–10, June 2016.

[LHS03] Qingsong Lu, Yan Huang, and Shashi Shekhar. Evacuation planning: a capacity constrained routing approach. In *Intelligence and Security Informatics*, pages 111–125. Springer, 2003.

[LS11] Z. Li and G. Simon. Time-shifted tv in content centric networks: The case for cooperative in-network caching. In *2011 IEEE International Conference on Communications (ICC)*, pages 1–6, June 2011.

[LZHH02] Suihong Liang, A Nur Zincir-Heywood, and Malcolm I Heywood. Intelligent packets for dynamic network routing using distributed genetic algorithm. In *GECCO*, pages 88–96, 2002.

[MAZ17] Spyridon Mastorakis, Alexander Afanasyev, and Lixia Zhang. On the evolution of ndnSIM: an open-source simulator for NDN experimentation. *ACM Computer Communication Review*, July 2017.

[MCG11] Luca Muscariello, Giovanna Carofiglio, and Massimo Gallo. Bandwidth and storage sharing performance in information centric networking. In *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ICN '11, pages 26–31, New York, NY, USA, 2011. ACM.

[Med10] Deepankar Medhi. *Network routing: algorithms, protocols, and architectures*. Morgan Kaufmann, 2010.

[Meh14] F. Mehdipour. Smart field monitoring: An application of cyber-physical systems in agriculture (work in progress). In *2014 IIAI 3rd International Conference on Advanced Applied Informatics*, pages 181–184, Aug 2014.

[MFR78] John M McQuillan, Gilbert Falk, and Ira Richer. A review of the development and performance of the ARPANET routing algorithm. *Communications, IEEE Transactions on*, 26(12):1802–1811, 1978.

[MKB+16] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda. Cyber-physical systems in manufacturing. *CIRP Annals*, 65(2):621 – 641, 2016.

[Moy97] John Moy. RFC 2178: OSPF version 2. *IETF, Jul*, 1997.

[MRR80] John M McQuillan, Ira Richer, and Eric C Rosen. The new routing algorithm for the ARPANET. *Communications, IEEE Transactions on*, 28(5):711–719, 1980.

[MTS98] Masaharu Munetomo, Yoshiaki Takai, and Yoshiharu Sato. A migration scheme for the genetic adaptive routing algorithm. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, volume 3, pages 2774–2779. IEEE, 1998.

[Mun97] Masaharu Munetomo. An adaptive network routing algorithm employing path genetic operators. In *Proc. of the Seventh Inter. Conf. on Genetic Algorithms, 1997*, 1997.

[MW77] John M McQuillan and David C Walden. The ARPA network design decisions. *Computer Networks (1976)*, 1(5):243–289, 1977.

[MWS14] Frank Müller, Oliver Wohak, and Andreas Schadschneider. Study of influence of groups on evacuation dynamics using a cellular automaton model. *Transportation Research Procedia*, 2:168–176, 2014.

[NFST15] D. Nguyen, M. Fukushima, K. Sugiyama, and A. Tagami. Efficient multipath forwarding and congestion control without route-labeling in ccn. In *2015 IEEE International Conference on Communication Workshop (ICCW)*, pages 1533–1538, June 2015.

[NS-] NS-3. Available: url=https://www.nsnam.org/.

[NSF] NSF Named Data Networking project. Available: url=http://www.named-data.net/.

[NSK+06] K. Nishinari, K. Sugawara, T. Kazama, A. Schadschneider, and D. Chowdhury. Modelling of self-driven particles: foraging ants and pedestrians. *Physica A*, 372:132–141, 2006.

[OJ10] Denis Ouzecki and Dragan Jevtic. Reinforcement learning as adaptive network routing of mobile agents. In *MIPRO, 2010 Proceedings of the 33rd International Convention*, pages 479–484. IEEE, 2010.

[Ora90] David Oran. RFC 1142: OSI IS-IS intra-domain routing protocol. 1990.

[OS12] Mohd Fauzi Othman and Khairunnisa Shazali. Wireless sensor network applications: A study in environment monitoring system. *Procedia Engineering*, 41:1204 – 1210, 2012. International Symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012).

[PAR+18] Ioannis Psaras, Onur Ascigil, Sergi Rene, George Pavlou, Alex Afanasyev, and Lixia Zhang. Mobile data repositories at the edge. *USENIX Workshop on Hot Topics in Edge Computing (HotEdge 18)*, Jul 2018.

[PB03] Stefan Podlipnig and Laszlo Böszörmenyi. A survey of web cache replacement strategies. *ACM Comput. Surv.*, 35(4):374–398, December 2003.

[PCP12] 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*, ICN '12, pages 55–60, New York, NY, USA, 2012. ACM.

[PD07] Larry L Peterson and Bruce S Davie. *Computer networks: a systems approach*. Elsevier, 2007.

[Per10] Charles E. Perkins. IP Mobility Support for IPv4, Revised. RFC 5944, November 2010.

[PJA11] C. Perkins, D. Johnson, and J. Arkko. Mobility support in ipv6. RFC 6275, RFC Editor, July 2011.

[PPJ11a] J. Pan, S. Paul, and R. Jain. A survey of the research on future internet architectures. *IEEE Communications Magazine*, 49(7):26–36, July 2011.

[PPJ11b] Subharthi Paul, Jianli Pan, and Raj Jain. Architectures for the future networks and the next generation internet: A survey. *Computer Communications*, 34(1):2 – 42, 2011.

[PRH17] D. Posch, B. Rainer, and H. Hellwagner. SAF: Stochastic adaptive forwarding in named data networking. *IEEE/ACM Transactions on Networking*, 25(2):1089–1102, April 2017.

[QPV01] Lili Qiu, V. N. Padmanabhan, and G. M. Voelker. On the placement of web server replicas. In *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, volume 3, pages 1587–1596 vol.3, 2001.

[QRWM13] H. Qian, R. Ravindran, G. Q. Wang, and D. Medhi. Probability-based adaptive forwarding strategy in named data networking. In *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, pages 1094–1101, May 2013.

[rfc81] Transmission Control Protocol. RFC 793, September 1981.

[RGS+15] Jaziar Radianti, Ole-Christoffer Granmo, Parvaneh Sarshar, Morten Goodwin, Julie Dugdale, and Jose J Gonzalez. A spatio-temporal probabilistic model of hazard-and crowd dynamics for evacuation planning in disasters. *Applied Intelligence*, 42(1):3–23, 2015.

[RK09] E. J. Rosensweig and J. Kurose. Breadcrumbs: Efficient, best-effort content location in cache networks. In *IEEE INFOCOM 2009*, pages 2631–2635, April 2009.

[RLH06] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), January 2006.

[Ros80] Eric C Rosen. The updating protocol of ARPANET's new routing algorithm. *Computer Networks (1976)*, 4(1):11–19, 1980.

[RPS16] H. K. Rath, B. Panigrahi, and A. Simha. On cooperative on-path and off-path caching policy for information centric networks (icn). In *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, pages 842–849, March 2016.

[RR11] Dario Rossi and Giuseppe Rossini. Caching performance of content centric networks under multi-path routing (and more). *Relatório técnico, Telecom ParisTech*, pages 1–6, 2011.

[RR12] D. Rossi and G. Rossini. On sizing ccn content stores by exploiting topological information. In *2012 Proceedings IEEE INFOCOM Workshops*, pages 280–285, March 2012.

[RR14] Giuseppe Rossini and Dario Rossi. Coupling caching and forwarding: Benefits, analysis, and implementation. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, ACM-ICN '14, pages 127–136, New York, NY, USA, 2014. ACM.

[Sak10] Georgia Sakellari. The cognitive packet network: A survey. *The Computer Journal*, 53(3):268, 2010.

[SDC97] Devika Subramanian, Peter Druschel, and Johnny Chen. Ants and reinforcement learning: A case study in routing in dynamic networks. In *IJCAI (2)*, pages 832–839. Citeseer, 1997.

[SE12] G. Ch. Sirakoulis and S. Bandini (Eds.). *Cellular Automata - 10th International Conference on Cellular Automata for Research and Industry, ACRI 2012*. LNCS 7495, Springer, 2012.

[SFP17] Susmit Shannigrahi, Chengyu Fan, and Christos Papadopoulos. Request aggregation, caching, and forwarding strategies for improving large climate data distribution with ndn: A case study. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ICN '17, pages 54–65, New York, NY, USA, 2017. ACM.

[SG18] Will Serrano and Erol Gelenbe. The random neural network in a neurocomputing application for web search. *Neurocomputing*, 280:123–134, 2018.

[SGFT13] V. Sourlas, L. Gkatzikis, P. Flegkas, and L. Tassiulas. Distributed cache management in information-centric networks. *IEEE Transactions on Network and Service Management*, 10(3):286–299, September 2013.

[SGS14] Eleftherios Spartalis, Ioakeim G. Georgoudas, and Georgios Ch. Sirakoulis. Ca crowd modeling for a retirement house evacuation with guidance. In Jarosław Wąs, Georgios Ch. Sirakoulis, and Stefania Bandini, editors, *Cellular Automata*, pages 481–491, Cham, 2014. Springer International Publishing.

[SK05] Karen Seo and Stephen Kent. Security Architecture for the Internet Protocol. RFC 4301, December 2005.

[SLYJ15] Sumanta Saha, Andrey Lukyanenko, and Antti Ylä-Jääski. Efficient cache availability management in information-centric networks. *Comput. Netw.*, 84(C):32–45, June 2015.

[SPFT09] V. Sourlas, G. S. Paschos, P. Flegkas, and L. Tassiulas. Caching in content-based publish/subscribe systems. In *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pages 1–6, Nov 2009.

[Spi85] John Michael Spinelli. Broadcasting topology and routing information in computer networks. Technical report, DTIC Document, 1985.

[SRS15] Divya Saxena, Vaskar Raychoudhury, and Nalluri SriMahathi. Smarthealth-ndnot: Named data network of things for healthcare services. In *Proceedings of the 2015 Workshop on Pervasive Wireless Healthcare*, MobileHealth '15, pages 45–50, New York, NY, USA, 2015. ACM.

[SRS+16] Divya Saxena, Vaskar Raychoudhury, Neeraj Suri, Christian Becker, and Jiannong Cao. Named data networking: A survey. *Computer Science Review*, 19:15 – 55, 2016.

[STJ+13] K. Suksomboon, S. Tarnoi, Y. Ji, M. Koibuchi, K. Fukuda, S. Abe, N. Motonori, M. Aoki, S. Urushidani, and S. Yamada. Popcache: Cache more or less based on content popularity for information-centric networking. In *38th Annual IEEE Conference on Local Computer Networks*, pages 236–243, Oct 2013.

[SYZZ16] Klaus Schneider, Cheng Yi, Beichuan Zhang, and Lixia Zhang. A practical congestion control scheme for named data networking. In *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, ACM-ICN '16, pages 21–30, New York, NY, USA, 2016. ACM.

[TAG+13] Stefano Traverso, Mohamed Ahmed, Michele Garetto, Paolo Giaccone, Emilio Leonardi, and Saverio Niccolini. Temporal locality in today's content caching: Why it matters and how to model it. *SIGCOMM Comput. Commun. Rev.*, 43(5):5–12, November 2013.

[TPT06] Yu-Chee Tseng, Meng-Shiuan Pan, and Yuen-Yung Tsai. Wireless sensor networks for emergency navigation. *Computer*, 39(7):55–62, 2006.

[Tra97] Inc TranSafety. Study compares older and younger pedestrian walking speeds, 1997.

[VMC13] G. Vizzari, L. Manenti, and L. Crociani. Adaptive pedestrian behaviour for the preservation of group cohesion. *Complex Adaptive Systems Modeling*, 1(7):1–29, 2013.

[WBG16] Lan Wang, Olivier Brun, and Erol Gelenbe. Adaptive workload distribution for local and remote clouds. In *2016 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2016, Budapest, Hungary, October 9-12, 2016*, pages 3984–3988, 2016.

[WF11] Patrick Wendell and Michael J. Freedman. Going viral: Flash crowds in an open cdn. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 549–558, New York, NY, USA, 2011. ACM.

[WFZ04] HorstF. Wedde, Muddassar Farooq, and Yue Zhang. Beehive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. In Marco Dorigo, Mauro Birattari, Christian Blum, LucaMaria Gambardella, Francesco Mondada, and Thomas StÃijtzle, editors, *Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *Lecture Notes in Computer Science*, pages 83–94. Springer Berlin Heidelberg, 2004.

[WG15a] Lan Wang and Erol Gelenbe. Demonstrating voice over an autonomic network. In *Autonomic Computing (ICAC), 2015 IEEE International Conference on*, pages 139–140. IEEE, 2015.

[WG15b] Lan Wang and Erol Gelenbe. Experiments with smart workload allocation to cloud servers. In *Network Cloud Computing and Applications (NCCA), 2015 IEEE Fourth Symposium on*, pages 31–35. IEEE, 2015.

[WG18] L. Wang and E. Gelenbe. Adaptive dispatching of tasks in the cloud. *IEEE Transactions on Cloud Computing*, 6(1):33–45, Jan 2018.

[WGMK11] W. Wong, M. Giraldi, M. F. Magalhaes, and J. Kangasharju. Content routers: Fetching data on network path. In *2011 IEEE International Conference on Communications (ICC)*, pages 1–6, 2011.

[WHY+12] Lan Wang, A K M Mahmudul Hoque, Cheng Yi, Adam Alyyan, and Beichuan Zhang. OSPFN: An OSPF based routing protocol for named data networking. Technical Report NDN-0003, NDN, July 2012.

[WLT+13] Yonggong Wang, Zhenyu Li, G. Tyson, S. Uhlig, and G. Xie. Optimal cache allocation for content-centric networking. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*, pages 1–10, Oct 2013.

[WRL+14] J. Wang, J. Ren, K. Lu, J. Wang, S. Liu, and C. Westphal. An optimal cache management framework for information-centric networks with network coding. In *2014 IFIP Networking Conference*, pages 1–9, June 2014.

[WWK12] W. Wong, Liang Wang, and J. Kangasharju. Neighborhood search and admission control in cooperative caching networks. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 2852–2858, Dec 2012.

[WWK13] Liang Wang, O. Waltari, and J. Kangasharju. Mobiccn: Mobility support with greedy routing in content-centric networks. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 2069–2075, Dec 2013.

[XVS+14] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos. A survey of information-centric networking research. *IEEE Communications Surveys Tutorials*, 16(2):1024–1049, Second 2014.

[YAM+13] Cheng Yi, Alexander Afanasyev, Ilya Moiseenko, Lan Wang, Beichuan Zhang, and Lixia Zhang. A case for stateful forwarding plane. *Comput. Commun.*, 36(7):779–791, April 2013.

[YAW+12] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive forwarding in named data networking. *SIGCOMM Comput. Commun. Rev.*, 42(3):62–67, June 2012.

[YS07a] Y. F. Yu and W. G. Song. Cellular automaton simulation of pedestrian counter flow considering the surrounding environment. *Physical Review E*, 75:046112, 2007.

[YS07b] YF Yu and WG Song. Cellular automaton simulation of pedestrian counter flow considering the surrounding environment. *Physical Review E*, 75(4):046112, 2007.

[ZAB+14] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, kc claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *SIGCOMM Comput. Commun. Rev.*, 44(3):66–73, July 2014.

[ZLZ15] M. Zhang, H. Luo, and H. Zhang. A survey of caching mechanisms in information-centric networking. *IEEE Communications Surveys Tutorials*, 17(3):1473–1499, thirdquarter 2015.

[ZYZ+18] 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. Technical Report NDN-0057, Revision 2, NDN, April 2018.

[ZZL09] Xiaoping Zheng, Tingkuan Zhong, and Mengting Liu. Modeling crowd evacuation of a building based on seven methodological approaches. *Building and Environment*, 44(3):437–445, 2009.

[ZZZ14] Yu Zhang, Hongli Zhang, and Lixia Zhang. Kite: A mobility support scheme for ndn. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, ACM-ICN '14, pages 179–180, New York, NY, USA, 2014. ACM.