Enhanced Forwarding Strategies in Information Centric Networking

submitted to the
Faculty of Physics and Electrical Engineering,
University of Bremen

for obtainment of the academic degree

Doktor-Ingenieur (Dr.-Ing.)

Dissertation

by

Asanga Udugama, M.Eng. BBA
from Kurunegala, Sri Lanka

First assessor:	Prof. Dr. rer. nat. habil. Carmelita Görg
Second assessor:	Prof. Dr. -Ing. Andreas Timm-Giel
Submission date:	27 January 2015
Colloquium date:	24 April 2015

I assure, that this work has been done solely by me without any further help from others except for the official support of the Communication Networks group of the University of Bremen. The literature used is listed completely in the bibliography.

Bremen, 27th of January 2015

(Asanga Udugama)

Dedication

I dedicate this work to my parents, late Gamini and Kumudu Udugama, my aunt, late Nelum Gunasekera and my wife, Koojana Kuladinithi for their unconditional love, care, unwavering support, kindness and foresight.

Acknowledgments

This doctorate is a culmination of the many years of work that I have done at the Communications Networks (ComNets) group, University of Bremen. I have worked for more than 12 years at ComNets and during this time, I had the privilege of working together with some of the smartest people around.

First and foremost, I thank Prof. Carmelita Görg for taking me into the ComNets fold and for being a great boss that many can only dream of having. Not only did she give me this opportunity to do a doctorate, but she also made all the arrangements for me to do my Masters at the Cork Institute of Technology (CIT), Cork, Ireland. I have gained immensely from her wisdom, her critical thinking and most importantly, from her pragmatism.

I thank Prof. Andreas Timm-Giel, who, as a former colleague and as a good friend, has been a source of many ideas for all the work that I did at ComNets, including this doctorate. The discussions we had and the valuable inputs given during our times together working on projects have tremendously assisted me in formulating concepts.

I thank Dr. Dirk Pesch for the opportunity offered to undertake my Masters at the CIT. Your ideas used when performing the research during the Masters and your considerateness in arranging the funding is greatly appreciated.

All the current and former colleagues of ComNets have been a great strength to me all throughout my stay at ComNets. They were there for me whenever I required assistance. They were a source of ideas, know-how and much more. I thank Dr. Niko Fikouras, Dr. Eugen Lamers, Dr. Andreas Könsgen, Dr. Yasir Zaki, Liang Zhao, Dr. Umar Toseef, Dr. Xi Li, Dr. Thushara Weerawardene, Dr. Markus Becker, Dr. Bernd-Ludwig Wenning, Amanpreet Singh, Thomas Pötsch, Karl-Heinz Volk and Martina Kammann for their support. Additionally, I thank the colleagues at ComNets, TUHH, Hamburg, especially Jonas Eymann and Yunqi Luo for their valuable inputs given during our discussions.

The burden that I would have had to carry would have been too much if it weren't for the students who helped me along the way through their thesis work, project work and student jobs. With their work, they opened up new areas that I could work on, to complete my doctorate. I thank Xinyi Zhang, Sameera Palipana and Jinglu Cai for their support.

Last, but not least, I thank Koojana and Hasini from the bottom of my heart. Koojana, my wife is a towering strength who provided ideas, encouragement, support and the peace of mind required to work on my doctorate. Her critical thinking abilities has immensely helped me to rethink and refine my ideas. Hasini, my daughter has had the kindness and the patience to tolerate my absence from the many paternal responsibilities, to complete my doctorate.

Asanga Udugama

Abstract

Network use has evolved over time to be dominated by content distribution and retrieval, while networking technology is mainly concerned with connections between named hosts. Accessing contents and services require the mapping of *what* users care about to *where* the content is located. Information Centric Networking (ICN) is a new paradigm in networking and a future Internet architecture which treats content as the primitive - decoupling location from identity, security and access - to retrieve content by name. Together with built-in capabilities for caching content, multi-path communications, disruption tolerance and security, ICN is able to leverage advancements in technology to address the issues associated with the mismatch between today's network use and network architecture. There are a number of ICN architectures, viz., Network of Information (NetInf), Publish-Subscribe Internet Routing Paradigm (PSIRP), Data Oriented Network Architecture (DONA) and Content Centric Networking (CCN) being developed by researchers worldwide with the ultimate focus of replacing the current Internet. CCN, a "Clean Slate" architecture to ICN, uses new approaches to routing named content, achieving scalability, security and performance.

Most computing devices of today (user devices as well as in the core network) are equipped with multiple networking interfaces and hence, have the ability to use multiple paths within or over different networks. The widest possible experience in using multi-path is mostly available in Internet Protocol (TCP/IP) based networks and the research done shows that multi-path routing can achieve robustness, load balancing, congestion reduction, low power consumption, and higher throughput. CCN, which has built-in capabilities to handle multiple networking attachments proposes two forwarding strategies, viz., the standard strategy which replicates network traffic into multiple paths and the best-face strategy which uses the discovered multiple paths one at a time, alternatively. However, these forwarding strategies do not exploit the full potential of CCN, especially the multi-path capability to improve performance in CCN based networks. The main objective of this thesis therefore, is to identify a set of mechanisms that capitalize on the multi-path capability of CCN to improve the overall performance of user applications and the deployed networks.

The contributions made by this thesis are twofold. Firstly, this thesis proposes a design of an effective multi-path forwarding strategy and performs an evaluation of this strategy in a set of scenarios that consider large scale deployments. This forwarding strategy, referred to as the On-demand Multi-path - Interest Forwarding (OMP-IF) strategy is an effective and a reliable set of mechanisms to discover, use and maintain simultaneously used multiple paths. The OMP-IF strategy creates node disjoint multiple paths making decisions dynamically on a hop-by-hop basis, improving reliability in communications and maximizing the benefits of bandwidth aggregation in stationary as well as mobile networking environments.

The performance of the OMP-IF strategy is evaluated in a simulation model developed for Optimized Network Engineering Tools (OPNET) in the course of this thesis work. The simulation model has the complete CCN protocol stack overlaid on User Datagram Protocol (UDP), included with the functionalities of flow and congestion control, client-server based CCN applications, Zipf based content popularity in applications, modeled and real Least Recently Used (LRU) caching, and with a random direction based mobility model. The results show improved performance with the proposed mechanisms in terms of user application throughput, delays, adoptability and scalability against adverse conditions (such as differing background loads and mobility) compared to the originally proposed forwarding strategies.

Secondly, this thesis proposes an analytical model to characterize multi-path data transfers in CCN. The model, which represents the CCN request arrivals using a Markov Modulated Rate Process (MMRP) extends an LRU based analytical model on caching to support multi-path content retrievals that use splitting of content requests to multiple paths to aggregate bandwidth. The performance of the analytical model is compared against the performance of the simulation model built in this thesis in scenarios that include single and multi level caches. The results show a close resemblance in performance between the analytical model and the simulation model.

The concepts, mechanisms and the analysis of results presented in this thesis are of great value for the success of adopting CCN as a viable future Internet architecture. The use of network operator based scenarios to evaluate the performance makes the case for using CCN in today's networks which are used mainly to move content. Though CCN is used as the basis for adopting and evaluating the proposed mechanisms and the analytical model, they are equally applicable and adaptable for use in other ICN architectures.

Kurzfassung

Die Nutzung von Netzen hat sich im Laufe der Zeit zunehmend auf das Verteilen und Abrufen von Inhalten verlagert, während bei der Netztechnologie weiterhin nur von Verbindungen zwischen benannten Hosts die Rede ist. Der Zugriff auf Inhalte und Dienste erfordert die Abbildung von dem, was Nutzern wichtig ist, zum Ort, an dem der Inhalt gespeichert ist. Information Centric Networking (ICN) ist ein neues Paradigma und eine zukünftige Internetarchitektur, die Inhalte als Primitive behandelt – wobei der Ort von Identität, Sicherheit und Zugriff entkoppelt wird – um Inhalte anhand ihres Namens abzurufen. Zusammen mit den integrierten Fähigkeiten zum Zwischenspeichern von Inhalten, Mehrpfade-Kommunikation, Störfestigkeit und Sicherheit ist ICN in der Lage, Weiterentwicklungen in der Technologie anzuwenden, um Fragestellungen in Zusammenhang mit der fehlenden Anpassung zwischen der heutigen Netznutzung und -architektur zu begegnen. Es gibt eine Anzahl von ICN-Architekturen, nämlich Network of Information (NetInf), Publish-Subscribe Internet Routing Paradigm (PSIRP), Data Oriented Network Architecture (DONA) und Content Centric Networking (CCN), die von Forschern weltweit mit dem Ziel entwickelt werden, das gegenwärtige Internet zu ersetzen. CCN, eine von Grund auf neuartige Lösung für ICN, verwendet neue Ansätze für das Routen benannter Inhalte, Erzielen von Skalierbarkeit, Sicherheit und Leistungsfähigkeit.

Die meisten heutigen digitalen Geräte (sowohl die Endnutzergeräte als auch die im Kernnetz) sind mit verschiedenen Netzschnittstellen ausgestattet und haben somit die Fähigkeit, mehrere Pfade über unterschiedliche Netze zu nutzen. Die umfangreichsten Erfahrungen in der Nutzung von Mehrpfadeübertragungen gibt es in TCP/IP-basierten Netzen, und die Forschungsergebnisse zeigen, dass Mehrpfaderouting Robustheit, Lastverteilung, Reduktion von Überlast, eine geringe Leistungsaufnahme und höheren Durchsatz erzielen kann. CCN, welches über integrierte Fähigkeiten zur Behandlung mehrerer Netzzugänge verfügt, sieht zwei Strategien für das Forwarding vor, nämlich die Standardstrategie, die den Netzverkehr mehrfach auf die verschiedenen Pfade verteilt, und die Best-Face-Strategie, die jeweils einen der ermittelten Pfade alternativ benutzt. Diese Forwarding-Strategien schöpfen jedoch das volle Potenzial von CCN nicht aus, insbesondere die Mehrpfade-Fähigkeit kann zur Verbesserung des Leistungsverhaltens in CCN-basierten Netzen beitragen. Das Hauptziel dieser Arbeit ist daher, eine Zusammenstellung von Mechanismen zu identifizieren, die aus den Mehrpfade-Fähigkeiten von CCN Nutzen ziehen, um die gesamte Leistungsfähigkeit von Nutzeranwendungen und den eingesetzten Netzen zu verbessern.

Die Beiträge dieser Arbeit sind zweierlei: Zum einen schlägt diese Arbeit den Entwurf einer effektiven Mehrpfade-Forwarding-Strategie vor und führt eine Evaluierung dieser Strategie in einer Anzahl von Szenarien durch, die einen Einsatz in großem Maßstab berücksichtigen. Diese Forwarding-Strategie, als On-demand Multipath Interest Forwarding (OMP-IF) bezeichnet, ist eine effektive und zuverlässige Zusammenstellung von Mechanismen für die Ermittlung, Nutzung und Steuerung mehrerer gleichzeitig genutzter Pfade. Die OMP-IF-Strategie erzeugt mehrere Pfade mit disjunkten Knoten, wobei Entscheidungen dynamisch auf einer Hop-zu-Hop-Basis getroffen werden, was die Zuverlässigkeit der Kommunikation erhöht und die Vorteile eine Bandbreitenaggregierung in stationären sowie in mobilen Netzumgebungen maximiert.

Das Leistungsverhalten der OMP-IF-Strategie wird in einem innerhalb dieser Arbeit für Optimized Network Engineering Tools (OPNET) entwickelten Simulationsmodell ausgewertet. Das Simulationsmodell enthält den vollständigen auf TCP/IP aufgesetzten CCN-Protokollstapel einschließlich der Funktionalitäten der Fluss- und Überlaststeuerung, Client-Server-basierten CCN-Anwendungen, Modellierung Zipf-basierter Popularität von Inhalten in Anwendungen, Least Recently Used (LRU)-Zwischenspeicherung und einem auf Zufallsrichtungen basierten Mobilitätsmodell. Die Ergebnisse zeigen ein mit den vorgeschlagenen Mechanismen verbessertes Leistungsverhalten bezogen auf den anwendungsbezogenen Durchsatz, Verzögerung, Anpassbarkeit und Skalierbarkeit unter ungünstigen Bedingungen (beispielsweise schwankende Hintergrundlast und Mobilität) verglichen mit den ursprünglich vorgeschlagenen Forwarding-Strategien.

Zum zweiten bringt diese Arbeit ein analytisches Modell für die Charakterisierung von Mehrpfade-Datenübertragungen in CCN ein. Das Modell, dass die Ankünfte von CCN-Anforderungen als einen Markov Modulated Rate Process (MMRP) beschreibt, erweitert ein LRU-basiertes analytisches Modell für die Zwischenspeicherung zur Unterstützung von Mehrpfade-Inhaltsabfragen, die die Aufteilung von Inhaltsabfragen auf mehrere Pfade für die Aggregierung der Bandbreite verwenden. Das Leistungsverhalten des analytischen Modells wird mit dem in dieser Arbeit erstellten Simulationsmodell verglichen, welches Zwischenspeicher auf einzelnen oder mehreren Ebenen einschließt. Die Ergebnisse zeigen eine enge Übereinstimmung der Ergebnisse zwischen dem analytischen und dem simulativen Modell.

Die in dieser Arbeit vorgestellten Konzepte, Mechanismen und Ergebnisanalysen sind von besonderem Wert für den Erfolg bei der Anpassung von CCN als eine machbare Architektur des zukünftigen Internets. Die Verwendung von Netzbetreiber basierten Szenarien für die Auswertung des Leistungsverhaltens liefert überzeugende Argumente bzgl. der heutigen Netzen, die hauptsächlich für die Übertragung von Inhalten verwendet werden. Auch wenn CCN als Grundlage für die Anpassung und Auswertung der vorgeschlagenen Mechanismen und des analytischen Modells verwendet wird, sind sie in gleicher Weise für die Nutzung in anderen ICN-Architekturen anwendbar und anpassbar.

Contents

Abstract **IX**

Kurzfassung **XI**

List of Figures **XXI**

List of Tables **XXV**

List of Abbreviations **XXVII**

List of Symbols **XXXI**

1 Introduction **1**

1.1 Information Centricity of Networking 1
1.2 Issues 3
1.3 Focus of Thesis 5
1.4 Outline of Thesis 8

2 Information Centric Networking **11**

2.1 Information and Naming 12
2.2 Caching and Storage 12
2.3 Routing 13
2.4 Security 14
2.5 ICN Architectures 14
2.5.1 Content Centric Networking (CCN) 14
2.5.2 Network of Information (NetInf) 15
2.5.3 Publish-Subscribe Internet Routing Paradigm (PSIRP) . . 16
2.5.4 Data Oriented Network Architecture (DONA) 17
2.6 Definitions of Terms 17
2.7 Content Centric Networking (CCN) 18
2.7.1 CCN Node Architecture 20
2.7.2 Flow Balance 22

2.7.3 Loop-free Forwarding 23
2.7.4 Forwarding Strategies 23
2.7.4.1 *Standard Strategy* 23
2.7.4.2 *Best-face Strategy* 23
2.7.5 Distributed Caching 24
2.7.6 Routing 25
2.7.7 CCN and TCP/IP 27
2.7.7.1 Architectural Relationship 27
2.7.7.2 Co-existence 27
2.7.8 CCN and Application Layer Caching Solutions 28

3 Multi-path Forwarding in CCN 29
3.1 Feasibility of Multi-path in CCN 29
3.1.1 CCN Does Not Require Explicit Discovery of Shortest Paths 30
3.1.2 Strategy Layer 32
3.1.3 Processes Involved in Multi-path 33
3.1.3.1 Discovery Process 33
3.1.3.2 Utilization Process 34
3.1.3.3 Maintenance Process 34
3.1.4 Nature of Multi-path in CCN 35
3.2 On-demand Multi-path - Interest Forwarding (OMP-IF) Strategy . 35
3.2.1 Design Goals 36
3.2.2 Strategy Layer Operations and Good Face List (GFL) . . . 37
3.2.3 Path Discovery 38
3.2.4 Path Utilization 42
3.2.4.1 Weighted Round Robin (WRR) 42
3.2.4.2 Leftover Capacity Utilization (LCU) 43
3.2.5 Path Maintenance 44
3.3 Influences on Caching 45
3.3.1 Cache Fragmentation 45
3.3.2 Cache Pollution 46
3.3.3 Persistence of Influences on Caching 46
3.4 Related Work on CCN Multi-path Forwarding Strategies 47
3.4.1 Caching Performance in Multi-path CCN 47
3.4.2 Stateful Forwarding Plane in NDN 48

4 Simulation Model 49
4.1 OPNET Environment 49
4.2 Protocol Stack 51

4.3 Application Layer 53
4.3.1 Client Application Process Model (*client_app*) 54
4.3.2 Server Application Process Model (*server_app*) 56
4.3.3 Traffic Generation Model 57
4.3.4 Flow and Congestion Control (F&CC) 59
4.3.4.1 F&CC Parameters 60
4.3.4.2 F&CC Algorithms 61
4.4 CCN Layer 64
4.4.1 CCN Process Model (*ccn_core*) 64
4.4.2 Forwarding Information Base (FIB) Population 66
4.4.3 Forwarding Strategies 66
4.4.3.1 OMP-IF Strategy 68
4.4.3.2 Best-face Strategy 68
4.4.4 Content Store (CS) Management 68
4.4.4.1 Cache Replacement Policy 69
4.4.4.2 Cache Decision Policy 69
4.4.4.3 Real Caching 69
4.4.4.4 Modeled Caching 71
4.4.5 Pending Interest Table (PIT) Management 71
4.4.6 Path Usage 73
4.4.7 Mobility Model 73
4.5 Adaptation Layer 76
4.5.1 Adaptation Layer Process Model (*ccnal*) 76
4.6 Transport and Network Layers 78
4.6.1 Transport Layer (UDP) 78
4.6.2 Network Layer (Internet Protocol (IP)) 78
4.7 Media Access Control (MAC) Layer 78

5 Performance Evaluation of Multi-path 81
5.1 Considered Scenarios 81
5.1.1 Network Based Multi-path Scenario (Fixed Network Operator (FNO) based Deployments) 81
5.1.2 Client Based Multi-path (Mobile Network Operator (MNO) based Deployments) 83
5.2 Simulation Environment 84
5.2.1 Network Topology 84
5.2.2 Popularity Model and Caching Configurations 85
5.2.3 Flow and Congestion Control (F&CC) Configurations . . 87
5.2.4 Pending Interest Table (PIT) Timeout Configurations . . . 88

5.2.5 Link Capacities 88
5.2.6 Mobility Parameters 89
5.2.7 Simulation Durations and Seeds 90
5.3 Simulation Cases 90
5.4 Evaluation Metrics 92
5.4.1 Macro Scale Metrics 93
5.4.1.1 Average Cache Hit Ratio 93
5.4.1.2 Average Content Download Time 93
5.4.1.3 Average Content Object Delay 94
5.4.1.4 Average Content Object Hop Count 94
5.4.1.5 Load Balancing Measure 95
5.4.1.6 Average Retrieval Ratio 95
5.4.1.7 Interest Flow Efficiency 96
5.4.2 Micro Scale Metrics 96
5.4.2.1 *cwnd* 96
5.5 Simulation Results and Analysis 96
5.5.1 Empty Caches vs. Pre-filled Caches 97
5.5.2 Modeled Caching with Flow and Congestion Control (F&CC) 98
5.5.3 Path Utilization 100
5.5.4 Performance in Fixed Network Operator (FNO) based Networks 102
5.5.4.1 Download Analysis - Single Content 106
5.5.4.2 Download Analysis - Zipf Classes 110
5.5.5 Performance in Mobile Network Operator (MNO) based Networks 111
5.5.6 Changes in Content Popularity 117

6 Analytical Characterization of Multi-path Content Retrievals 119
6.1 Related Work 119
6.1.1 Content Popularity and Topology 121
6.1.2 Content Size 121
6.1.3 Arrival Process 122
6.1.4 Cache Miss Probability 123
6.2 Analytical Model 124
6.2.1 Topology 124
6.2.2 Multi-path Usage Strategy 124
6.2.3 Multi-Path Model 125
6.2.3.1 Arrival Process 126
6.2.3.2 Cache Miss Probability 128

6.2.3.3 Second Level Cache Miss Probability 130
6.2.4 Performance Metrics 132
6.2.4.1 VRTT 132
6.2.4.2 Throughput 132
6.2.4.3 Download Times 133
6.3 Simulation Model 133
6.3.1 CCN Application Layer 134
6.3.2 CCN Layer 134
6.4 Validation of the MP Model 136
6.5 Performance Results and Analysis 137
6.5.1 VRTT 139
6.5.2 Throughput 140
6.5.3 Download Times 141
6.5.4 Overall Observations 142

7 Summary and Conclusions 145
7.1 Summary 145
7.2 Observations and Conclusions 147
7.3 Outlook 149

Appendix 151

A Modeled Caching in Simulator 153
A.1 Hit Ratio 153

B Statistical Estimations and Numerical Results 157
B.1 Statistical Evaluation: An Example 157
B.2 OMP-IF Evaluation Results 158
B.2.1 FNO Scenario: Average Download Time and Average Content Object Delay 159
B.2.2 FNO Scenario: Average Hop Count and Load Balancing Measure 159
B.2.3 FNO Scenario: Average Retrieval Ratio 160
B.2.4 MNO Scenario: Average Download Time and Average Content Object Delay 160
B.2.5 MNO Scenario: Average Hop Count and Load Balancing Measure 160
B.3 Analytical Model Evaluation Results 161
B.3.1 Hit Ratio 161

B.3.2 VRTT 162
B.3.3 Throughput 163
B.3.4 Download Time 165

Bibliography **167**

List of Figures

1.1 Extracting Value from Chaos, IDC Report, June 2011 2
1.2 Evolution of Network Use: From *where* to *what* 3
1.3 Forwarding Strategies Proposed by [JST^{+}09] 4

2.1 CCN Operations (Source: [ADI^{+}11]) 15
2.2 NetInf Operations (Source: [ADI^{+}11]) 15
2.3 PSIRP Operations (Source: [ADI^{+}11]) 16
2.4 DONA Operations (Source: [ADI^{+}11]) 16
2.5 CCN Terms 17
2.6 Protocol Stack Comparison of TCP/IP and CCN - Change of the "Thin Waist"from IP packets to named Content Chunks (Source: [JST^{+}09]) 19
2.7 CCN Message Formats (Source: [JST^{+}09]) 20
2.8 CCN Node Architecture (Source: [JST^{+}09]) 22
2.9 Conventional Content Retrievals (left) and Content Retrievals with Distributed Caching (right) 25
2.10 Populating the FIB and Forwarding in CCN 26

3.1 Link-State Advertisement (LSA) Protocol in TCP/IP vs. FIB Population in CCN 30
3.2 Operation of Strategy Layer 33
3.3 Structure of GFL 37
3.4 Selection of *good faces* 38
3.5 States and Transitions of GFL Entries 39
3.6 Node Disjoint Path Discovery - Interest Flow 40
3.7 Node Disjoint Path Discovery - Content Object Flow 40
3.8 Path Delay Asymmetry may Result in Disjoint Path Discovery Failures 41
3.9 WRR - Weights are used to distribute Interests 42
3.10 LCU - Face through which the last Content Object was received is used to send the next Interest 44

3.11 Effects on Caching - (a) Normal Cache, (b) Fragmented Cache, (c) Polluted Cache . . . 45

4.1 OPNET Editors: Network Model Editor, Node Model Editor, Process Model Editor and the Process Model Code Editor . . . 50
4.2 CCN Protocol Stack in Nodes . . . 52
4.3 A cross section of an OPNET scenario showing an inter-connected CCN Client (*ccn_client_1*) and CCN Routers . . . 52
4.4 Node Model of CCN in OPNET identified by Protocol Layer . . . 53
4.5 Finite State Machine (FSM) of the Client Process Model . . . 54
4.6 FSM of the Server Process Model . . . 55
4.7 Zipf popularities for different α values . . . 57
4.8 Adapted Flow and Congestion Control (F&CC) Algorithms . . . 64
4.9 FSM of the CCN Layer Process Model . . . 65
4.10 Interest Processing (left) and Content Object Processing (right) Flow Charts of best-face and OMP-IF Forwarding Strategies . . . 67
4.11 Content Coverage in a Cache . . . 70
4.12 Operation of the Soft-Hard PIT Timeout Mechanism . . . 72
4.13 Random Direction based Mobility Model . . . 74
4.14 FSM of the CCN Adaptation Layer Process Model . . . 77

5.1 CCN Deployment on Stationary Networks operated by FNOs . . . 82
5.2 CCN Deployment in Mobile Networks . . . 83
5.3 OPNET Scenario for Network based Multi-path with CCN . . . 85
5.4 Configuration of Cache Categories . . . 86
5.5 Coverage Map of MNO Scenario . . . 90
5.6 Comparison of Performance with Empty and Pre-filled Caches . . 98
5.7 Comparison of Performance with Modeled and Simulated Caching 99
5.8 Congestion Window (*cwnd*) Performance of a Single CCN Client Node . . . 99
5.9 Comparison of Performance with LCU and WRR . . . 100
5.10 Congestion Window (*cwnd*) Performance of a Single CCN Client Node . . . 101
5.11 Performance of Download Time and Content Object Delay in FNO Scenario . . . 102
5.12 Performance of Hop Count and Load Balancing in FNO Scenario . 104
5.13 Performance of Retrieval Ratio and Interest Flow Efficiency in FNO Scenario . . . 105

5.14 Progress of a Single Download of a Popular Class with Best-face Strategy in the FNO Scenario 107
5.15 Progress of a Single Download from a Popular Class with OMP-IF Strategy in the FNO Scenario 108
5.16 Progress of a Single Download from an Unpopular Class with Best-face Strategy in the FNO Scenario 109
5.17 Progress of a Single Download from an Unpopular Class with OMP-IF Strategy in the FNO Scenario 110
5.18 Per Class Performance for 30% BTL in the FNO Scenario 111
5.19 Face Connectivity Status in MNO Scenario of a Selected CCN Client Node 112
5.20 Performance of Download Time and Content Object Delay in MNO Scenario 113
5.21 Performance of Hop Count and Load Balancing in MNO Scenario 114
5.22 Performance of Face Connectivity in the MNO Scenario 115
5.23 Performance of Download Time in the MNO Scenario with Extended Connectivity Coverage (EC refers to Extended Coverage) . . 116
5.24 Performance of the Hit Ratio with Popularity Renewal (PR) of two Contents, each belonging to two different Zipf Classes 118

6.1 Single Cache Scenario 121
6.2 Cache miss probabilities for different α values 123
6.3 General Topology 125
6.4 Abstract Content Request Processes of a Client and at the Cache . 126
6.5 Markov Chain for the MMRP Process at a Cache 127
6.6 Simulated Single path (a) and Multi-path (b) Topologies 136
6.7 Hit Ratio Comparison for All Classes (Simulated and Analytical with Single Level of Caches) 137
6.8 Hit Ratio Comparison for Selected Classes (Simulated and Analytical with Single Level of Caches) 138
6.9 VRTT Comparison (Simulated and Modelled) 139
6.10 Comparison of Modelled and Simulated Throughput 140
6.11 Comparison of Modelled and Simulated Download Times 141

A.1 Hit Ratio Comparison for Modelled and Real Cache Implementations in Simulator 154

List of Tables

4.1 Parameters to Determine Content Coverage 70
4.2 Events and Actions of the Soft-Hard PIT Timeout Mechanism . . 72

5.1 Popularity Model Parameters Used in Simulations 86
5.2 Initial Values of F&CC Variables 88
5.3 PIT Timeout Values Used in Simulations 88
5.4 Link Capacities Used in Scenarios 89
5.5 Mobility Parameters used by CCN Client Nodes 89
5.6 Parameter Tables used in the Scenarios 97
5.7 Performance of Download Time in FNO Scenario - Numerical Results 103

6.1 Parameter Values Used in the Analytical and Simulated Models . . 135
6.2 Scenarios Considered and the Legend Identifications 138

7.1 Summary of Average Download Time Performance in FNO based Networks 148

B.1 Computation of Mean and Confidence Interval (CI) Values 158

List of Abbreviations

3GPP	3^{rd} Generation Partnership Project
ACK	TCP Acknowledgement Message
ccnal	Adaptation Layer Process Model
ALWAYS	Cache Always
AODV	Ad hoc On-Demand Distance Vector
ARQ	Automatic Repeat Request
APR	Alternate Path Routing
ARP	Address Resolution Protocol
arp	ARP Process Model
ccn_ app	Application Management Process Model
BFL	Best Face List
BTL	Background Traffic Load
CI	Confidence Interval
CoV	Coefficient of Variation
CCN	Content Centric Networking
CDN	Content Distribution Network
CS	Content Store
CDF	Cumulative Distribution Function
ccn_ core	CCN Process Model
CBR	Constant Bit Rate
client_ app	Client Application Process Model
CloNe	Cloud Networking
DONA	Data Oriented Network Architecture
DE	Decision Making Entity
DNS	Domain Name System
eth_ rx	Ethernet Receiver Process Model
eth_ tx	Ethernet Transmitter Process Model
EE	Execution and Enforcement Entity
FIB	Forwarding Information Base
FNO	Fixed Network Operator
FIX(P)	Random with a Fixed Probability
FIFO	First In First Out
FSM	Finite State Machine
F&CC	Flow and Congestion Control
FRT	Fast Retransmit Threshold
GFL	Good Face List
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICN	Information Centric Networking
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6

IS-IS	Intermediate System to Intermediate System
IGP	Interior Gateway Protocol
ip_ encap	IP Encapsulation Process Model
ip	IP Process Model
ISP	Internet Service Provider
IE	Information Management Entity
LSR	Link State Routing
LSA	Link-State Advertisement
LCU	Leftover Capacity Utilization
LRU	Least Recently Used
LFU	Least Frequently Used
LTE	Long Term Evolution
LPM	Longest Prefix Matching
LAN	Local Area Network
MMRP	Markov Modulated Rate Process
MAC	Media Access Control
MNO	Mobile Network Operator
mac	MAC Process Model
NetInf	Network of Information
NDN	Named Data Networking
NRS	Name Resolution Service
NACK	Negative Acknowledgment
nnrp	NEC NetInf Router Platform
OSPF	Open Shortest Path First
OSI	Open Systems Interconnection
OMP-IF	On-demand Multi-path - Interest Forwarding
OPNET	Optimized Network Engineering Tools
OConS	Open Connectivity Services
PSIRP	Publish-Subscribe Internet Routing Paradigm
PIT	Pending Interest Table
PARC	Palo Alto Research Center
PR	Popularity Renewal
P2P	Peer-to-Peer
PA	Prefix Advertisement
PoP	Point of Presence
RR	Round Robin
RTT	Round Trip Time
RI	Rendezvous Identifier
RH	Resolution Handler
RTO	Re-transmission Timeout
RAM	Random Access Memory
RNG	Random Number Generator
SDN	Software Defined Networking
SNR	Signal-to-Noise Ratio
SUM	Simultaneous Use of Multi-path
SL	Strategy Layer
SHPTM	Soft-Hard PIT Timeout Mechanism
server_ app	Server Application Process Model
SAIL	Scalable and Adaptive Internet Solution
TCP/IP	Internet Protocol
TCP	Transmission Control Protocol
TLV	Type-Length-Value
UNIF	Uniformly Distributed
UMTS	Universal Mobile Telecommunication System
udp	UDP Process Model
URL	Universal Resource Locator
UDP	User Datagram Protocol

uccn	User-space CCN
VRTT	Virtual Round Trip Time
WWW	World Wide Web
WAN	Wide Area Network
WLAN	Wireless Local Area Networking
WRR	Weighted Round Robin
XML	Extensible Markup Language

List of Symbols

Symbol	Meaning	Type
α	Exponent characterizing the Zipf distribution	Constant
c	Normalization constant	Constant
δ_k	Service rate of a cache for the class k	Mean
Γ	Gamma function	Constant
h_k	The hit ratio of the class k at the cache level i	Mean
H_k	Mean total number of hits generated at all caches for class k	Mean
K	Total number of content classes	Constant
k	Class number	Constant
λ_k	Content request rate of class k	Mean
$\lambda(i)$	Overall content request arrival rate at the class level i	Mean
$\lambda, \lambda(1)$	Overall content request arrival rate at the 1^{st} cache level	Mean
m	Number of contents in a class	Constant
$\mu(i)$	Overall content miss rate at the cache level i	Mean
N	Number of cache levels in the network	Constant
N_{ch}, a	Mean number of requests for chunks generated until no chunk is received	Mean
ω	Mean number of chunks in a content	Mean
p_l	Probability of an event occurring where no chunk is received, to identify the size of a content	Random Variable
$p_k(i)$	Cache miss probability at cache level i for the class k	Random Variable

Symbol	Meaning	Type
p_k, $p_k(1)$	Cache miss probability at the fist level cache for class k	Random Variable
q_k	Distribution of content associated with class k, i.e., the Zipf distribution denoted by $\frac{c}{k^\alpha}$	
R_i	Round trip delay between the user and the cache i	Constant
TR_k	Mean total request arrivals at all caches for class k	Mean
$VRTT_k$	Mean virtual round trip time of class k	Mean
W	Window size maintained at the client	Constant
x	Cache size in chunks	Constant
X_k	Throughput for class k	Mean
Y	Number of users in the network	Constant
Z	Number of caches in the network	Constant

1 Introduction

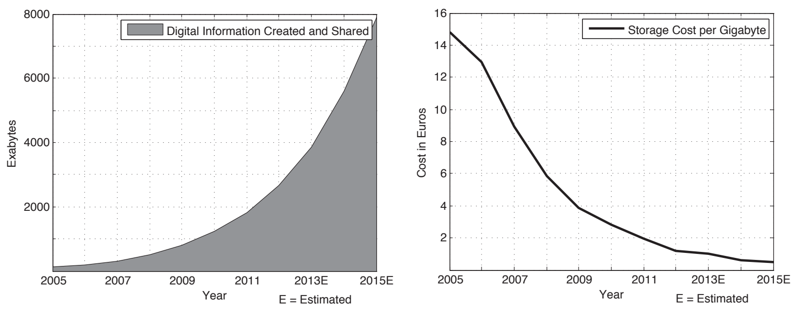
The architecture of today's computer networks had its origins in the late '60s and the early '70s. The purpose of networking at the time was to share computing resources. Computers built during this time were extremely expensive. Moreover, the peripherals such as printers and disks that were required to productively use the computers were equally expensive. Therefore, acquiring a peripheral device and making it available for the many users of a computer and even for the computers installed in different geographical locations was considered as an alternative to buying and attaching a peripheral for every computer.

To share these peripherals, it was required to draw cables between the computers and to develop software that was able to let users of other computers use the peripherals connected to other computers. These cables and the related software were considered as being components of a computer which was termed computer networking. There were many advances in the area of computer networking which resulted in the standardization of how computers communicated with these standards being referred to as networking protocols. These standards took a layered approach to defining protocols based on the idea of compartmentalizing the functionality. The Open Systems Interconnection (OSI) model [Tan02] is the outcome of the definition of the functionality associated with each layer. With the development of the Internet Protocol (TCP/IP) suite, a derivative of the OSI model with specific protocols handling functionality at each layer, and the development of the Internet and the World Wide Web (WWW), has resulted in networking being used not only to share physical resources but also to share information all over the world.

1.1 Information Centricity of Networking

With the use of computing devices in every aspect of the daily lives of people, bits are created and disseminated in huge volumes. Devices such as Smart Phones, Tablet Computers with powerful means of network connectivity such as Long Term Evolution (LTE) and services such as Facebook, Twitter, Youtube are enabling users to produce information and make them available instantly to be consumed by other users of such devices and services. Therefore, the purpose of today's network use has moved from a resource sharing objective to an information dissemination focus.

Figure 1.1(a) shows the growth of content created and shared, and how it is projected to grow in the future [GR11]. Though the focus of networking has changed to one of being information centric, the underlying networking architecture is still based on the architecture that was meant for the requirements of the '60s and '70s. This mismatch of networking architecture and network use has resulted in such issues as performance bottlenecks, data transmission deficiencies and data insecurities.



(a) Growth of the Digital Universe (b) Storage Cost Benefits

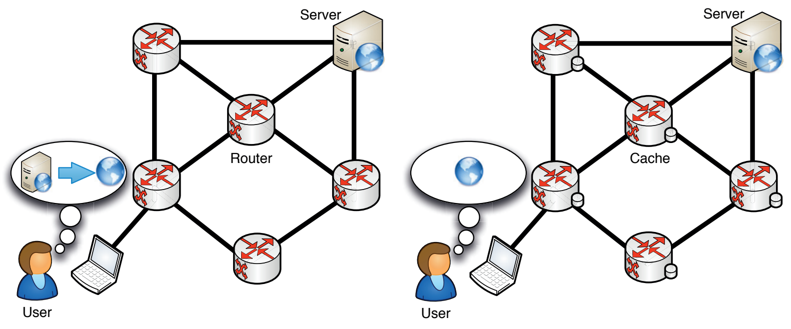
Figure 1.1: Extracting Value from Chaos, IDC Report, June 2011

In order to prevent current networks from grinding to a halt, a number of information aware application layer solutions such as Content Distribution Networks (CDNs), Peer-to-Peer (P2P) overlays and Hypertext Transfer Protocol (HTTP) Proxies have been devised to ease the strain on current networks. Yet, these approaches are overlaid on top of the current networking architecture without addressing the fundamental issues associated with the current architecture. Therefore, considering the futility of evolving the current architectures, there are a number of efforts to develop "Clean Slate" solutions that address the issue of providing a networking architecture suitable for the information centricity of today's network use.

One of these efforts is the architecture proposed by Palo Alto Research Center (PARC) called Content Centric Networking (CCN) [JST$^+$09] and subsequently further developed by the Named Data Networking (NDN) project [ZAB$^+$14]. The CCN architecture treats content as a primitive, choosing to name the contents rather than the hosts on which the content resides. It directly routes named contents at the packet level of a network enabling automatic and application independent caching in the network, taking advantage of the improvements and gains in storage technology (see Figure 1.1(b)) over the use of network bandwidth.

1.2 Issues

A user requesting contents in CCN based networks should only specify "*what*" contents are required rather than "*where*" the contents are located (Figures 1.2(a) and 1.2(b)). Every CCN node receiving these requests guides them to a cache that holds the requested content. The guiding of requests is performed by a forwarding engine located in each of the CCN nodes deployed with a strategy to leverage the key advantages of the CCN architecture, viz., named data (i.e., contents) and in-network caching.

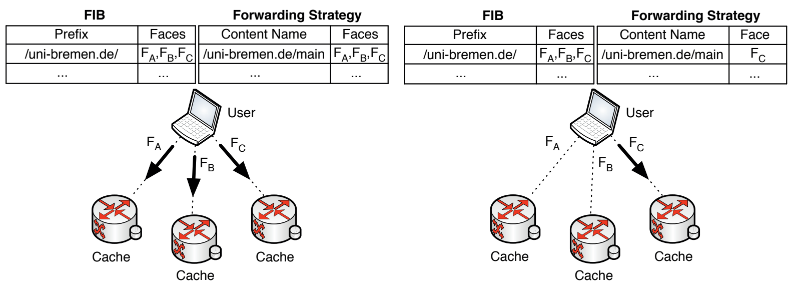


(a) Traditional Networking (b) Content Centric Networking (CCN)

Figure 1.2: Evolution of Network Use: From *where* to *what*

The fundamental operations of CCN is handled using 2 message types and 3 data structures. The message type *Interest* is used to request for a part of a content and this requested part of the content is sent using a *Content Object* message by the content provider or a cache. The Forwarding Information Base (FIB) data structure is used to store the forwarding information in terms of the content name prefixes and the network attachments (called Faces) through which the contents are reached. The Pending Interest Table (PIT) data structure holds the currently pending Interests that have not yet been served. PIT entries get created when an Interest propagates in a network. The Content Store (CS) data structure caches the Content Objects which have traversed a CCN node. [JST+09] proposes 2 strategies (also referred to as Forwarding Strategies) that the forwarding engine of a CCN node can use to forward the Interests in the direction of where the contents reside.

A CCN node deployed with the *Standard* forwarding strategy broadcasts copies of a received Interest to all the Faces given in a selected FIB entry (Figure 1.3(a)). The selected FIB entry is found by performing a Longest Prefix Matching (LPM) search using the prefix of the content name requested by the Interest. The *Best-face* forwarding strategy does the same (i.e., Interest replication), but only at the beginning of a content download and then, identifies a single Face to forward the subsequent Interests (Figure 1.3(b)). The Face selection is done by identifying the Face that brought back the first Content Objects.


FIB

Prefix	Faces
/uni-bremen.de/	F_A,F_B,F_C
...	...

Forwarding Strategy

Content Name	Faces
/uni-bremen.de/main	F_A,F_B,F_C
...	...

(a) *Standard* Forwarding Strategy

FIB

Prefix	Faces
/uni-bremen.de/	F_A,F_B,F_C
...	...

Forwarding Strategy

Content Name	Face
/uni-bremen.de/main	F_C
...	...

(b) *Best-face* Forwarding Strategy

Figure 1.3: Forwarding Strategies Proposed by [JST+09]

The operation of these 2 forwarding strategies results in a CCN application receiving Content Objects from the closest caches. These strategies, therefore are ideally suited to operate in lightly loaded networks, increasing the performance of the CCN applications. But, if the applications require a higher bandwidth or if the networks are highly loaded, these forwarding strategies impact negatively on performance as the required bandwidth is unable to be obtained.

Replication of Interests with the *Standard* strategy results in flooded networks which in turn increases the congestion and packet losses experienced by all applications. Using a single path (i.e., *Best-face* strategy) which is considered as being the best path to use at the beginning of a content download may become congested subsequently due to the use of the path and further, the strategy results in a skewed distribution of traffic over the network. In highly dynamic networks, such as with mobility, a single path scheme as in the *Best-face* strategy may result in higher packet drops and frequent path discoveries.

These issues therefore necessitate the investigation into more efficient and reliable forwarding mechanisms for CCN going beyond the forwarding strategies and the related mechanisms proposed in [JST+09]. This thesis investigates the utilization of alternative mechanisms centered around the use of multiple paths to distribute Interest traffic to avoid network flooding and enabling efficient and robust delivery of contents in CCN based networks.

1.3 Focus of Thesis

The use of multiple paths to route data in a network is an alternative to using single path routing. There are a number of aspects that makes the CCN architecture ideally suitable to perform multi-path communications. In the following is a discussion of these aspects.

- Previous research, mostly in the realm of TCP/IP based networks point to multi-path use as being beneficial to achieving robustness, load balancing, congestion reduction, low power consumption and higher throughput in networks [MGLA96, PP03, GGSE01, YKT03]. Further, multi-path routing is widely used in wireless networks to increase the robustness of communications due to frequent topology changes.

- The work done in [DBM11] concludes that the occurrence of creation and removal of content is more frequent than the creation and removal of links in today's networks. Therefore, when considering Information Centric Networking (ICN) based networks where routing is performed using names, ensuring the stability of routing is more difficult compared to conventional routing. Use of multi-path forwarding rather than single path forwarding is expected to be more helpful in environments with highly dynamic caching enabled network topologies.

- TCP/IP networks have no way of detecting loops in paths. Therefore, it requires additional support to build multiple paths. In CCN, every Interest carries a randomly generated nonce value to detect and discard duplicates [JST+09]. Further, the operations of CCN stipulate that any Interest that was previously received should not be forwarded again, but aggregated into the same PIT entry. In this manner, CCN avoids the establishment of path loops in a network and thereby, preventing the looping of Interests. The responses to Interests that carry the data i.e., Content Objects, follow the path set by PIT entries to the original requester of a content. If there is no matching PIT entry, a Content Object is simply discarded thereby preventing the establishment of looped paths with Content Objects. Therefore, as there is neither the danger of loops forming nor any restrictions on multi-source or multi-destination communications to avoid the establishment of loops, CCN is able to take full advantage of using multiple network attachments and multiple paths between the requesters and the sources of contents.

- The distributed in-network caching architecture in CCN naturally favors multi-point to multi-point communications. CCN allows every node that acts as a cache to cache the Content Objects traversing it. Therefore, in CCN unlike in TCP/IP networks, the Interests do not have to be routed to a single source since multiple CCN cache nodes are able to serve the same content.
- The routing layer of the CCN protocol stack differs from that of TCP/IP as it is more "intelligent" compared to TCP/IP. In TCP/IP, a packet received is forwarded to a single next hop after performing a LPM on the routing table to find the relevant next hop Internet Protocol (IP) address. CCN, on the other hand has the ability, in addition to the LPM search, to use other mechanisms to make decisions on how an incoming packet is forwarded. Therefore, CCN is able to maintain a dynamic forwarding strategy where each CCN node makes decisions per packet hop-by-hop on how Interests are forwarded. In this manner, a CCN node's forwarding strategy is able to take many factors such as delays, previous performance statistics, etc. into consideration in its decision process. Software Defined Networking (SDN) is an alternative considered in TCP/IP to incorporate more intelligence in the forwarding process. Though SDN is able to use different criteria to make forwarding decisions, it is still hampered by the non-ICN nature of TCP/IP.

As explained above, the CCN architecture has capabilities to perform multi-path communications. But, the exploitation of this capability is hindered due to the way the proposed forwarding strategies operate. Therefore, the purpose of this thesis is to identify a set of mechanisms that capitalize on the multi-path capability of CCN to improve its performance. This thesis makes the following contributions in the area of CCN.

- Developed and evaluated a unique CCN based multi-path establishment, use and maintenance mechanisms that is based on node disjoint path discovery, network conditions based path usage and a path maintenance mechanism with minimum disruptions to active downloads. This forwarding strategy is called the On-demand Multi-path - Interest Forwarding (OMP-IF) strategy.

- Built a simulation model that implements functionality to handle the complete CCN protocol stack overlaid on TCP/IP. The functionality consists of Flow and Congestion Control (F&CC) and client-server based applications, Zipf based content popularity in applications and caching, Least Recently Used (LRU) based modeled and real caching, random direction based mobility model, a CCN specific novel PIT management mechanism, a FIB population algorithm and multiple forwarding strategies that include the strategies proposed by the creators of CCN [JST$^+$09] and the unique forwarding strategy proposed in this work.

- Identified and used large scale evaluation scenarios applicable to network operators focusing on network based multi-path and client based multi-path implemented in the simulator using a network topology of AT&T Inc.

- Identified a set of macro and micro scale evaluation metrics relevant for evaluating the performance of large scale CCN deployments.

- Analyzed the performance comparing the original forwarding strategy proposed in [JST$^+$09] and the proposed OMP-IF in stationary and mobile networks.

- Performance analysis focussed on a number of aspects related to CCN which included the analysis on modeled vs. real caching, empty vs. pre-filled caches, best path utilization mechanism to use and content popularity renewals.

- Proposed and validated an analytical model to characterize multi-path content retrievals in CCN. The content arrivals which are characterized as a Markov Modulated Rate Process (MMRP) with an LRU based model to characterize caching is validated using the simulation model built in this thesis work.

In addition to the above mentioned work, in the following is a succinct description of other work done (not detailed in this thesis) related to multi-path in ICN based networks. The work relates to developing proof-of-concept implementations for EU FP7 Scalable and Adaptive Internet Solution (SAIL) Project. The SAIL Project consisted of 3 main work packages focused on ICN, Open Connectivity Services (OConS) and Cloud Networking (CloNe). As part of the OConS work package, multi-path mechanisms for ICN based networks were developed as concepts and implemented in a prototype for evaluations and demonstrations. Aspects of this research is described in [UGTG13] and [ZZU$^+$11].

- Light Weight CCN Implementation: A lightweight CCN prototype implementation called User-space CCN (uccn) was developed according to the functionality detailed in [JST$^+$09]. The purpose of the implementation was to include the multi-path content retrieval functionality developed in the OConS work package. The underlying transport was TCP/IP and used multihoming in TCP/IP to realize multiple CCN Faces. The multi-path utilized parts of the OMP-IF strategy that included splitting of the Interest flow and delay based weight determination to select the multiple Faces.
- Multi-path Extensions to Network of Information (NetInf): NetInf was the primary ICN architecture considered for research in the SAIL Project. NEC NetInf Router Platform (nnrp) implemented the NetInf concepts developed in the ICN work package. OConS, as a connectivity service to enable multi-path content retrievals for NetInf, was used to extend the nnrp implementation to include modules that performed multi-path. The OConS architecture with the Information Management Entity (IE), Decision Making Entity (DE) and Execution and Enforcement Entity (EE) which were responsible for managing the multi-path operations.

1.4 Outline of Thesis

The following chapters of this thesis present the research done in developing the multi-path mechanisms and the analytical model. The following text provides an overview to each of these chapters.

Chapter 2 provides an overview of the different ICN architectures in existence today, subsequently detailing the CCN architecture which is used in this thesis as the underlying ICN architecture to evaluate the proposed mechanisms. Before these explanations, a description is provided on the features that exist in any ICN architecture.

Chapter 3 provides the details of the multi-path mechanisms proposed in this thesis. The chapter begins with a discussion of the feasibility of multi-path in CCN and moves on to discuss the proposed multi-path mechanisms, collectively called the OMP-IF strategy. The last sections of this chapter discusses the influences that multi-path has on caching and a discussion on related multi-path work.

Chapter 4 provides a detailed description of the simulation model built to evaluate the multi-path mechanisms and the analytical model. The chapter begins with a brief overview of the Optimized Network Engineering Tools (OPNET), subsequently detailing the functionality developed to handle each layer of the CCN protocol stack.

Chapter 5 details the performance evaluation of the multi-path mechanisms proposed in this thesis. The chapter begins with a description of the scenarios used for evaluations, the simulation environment details such as network topology, popularity model parameters, etc. and the evaluation metrics used. The final sections of this chapter provides an analysis of the proposed mechanisms.

Chapter 6 provides the details of the analytical model developed to characterize multi-path content retrievals in CCN. The chapter begins by discussing the characteristics of the single path model which is extended to multi-path in this thesis. The second section details the multi-path analytical model developed in this thesis. The final section compares the performance evaluation of the analytical model against the performance of the simulation model.

Chapter 7 provides a summary of the work done, conclusions made and the extendable areas of the work presented.

2 Information Centric Networking

Information Centric Networking (ICN) came about due to the mismatch that exists in the use of current networks and the architecture of current networks. The architecture of today's networks is from the late '60s and early '70s where the focus of networking was resource sharing. But, these network architectures of the past are being used today primarily to move content from one location to another.

The realization of this mismatch brought about the creation of a number of global Future Internet research activities to develop concepts to rethink the focus of networking and one of these is to have an information orientation to networking. The main paradigm in ICN is the consideration of information or content as the primary entity rather than the host on which the information is stored as in current networks. Therefore, the focus is not on end-to-end communications between hosts but on network architectures based on highly scalable and efficiently distributed content. This has resulted in the emphasis being placed on information objects, the attributes of these objects and the requester of these information objects to conceptualize architectures that achieve efficient and reliable distribution of information objects. Thereby, these network architectures can leverage advantages such as in-network storage, multi-point interactions based on information replication and communication models such as publish-subscribe to enable an environment to implement communication services which are currently realized through Peer-to-Peer (P2P) overlays and proprietary Content Distribution Networks (CDNs).

As indicated before, ICN based networks have inherent capabilities to perform multi-path communications to retrieve content. Since most computing devices of today are equipped with multiple networking technologies, ICN architectures have the ability to use multiple paths over different networks when requesting content to gain many advantages. The widest possible experience in using multi-path is mostly available in Internet Protocol (TCP/IP) based networks and the research done shows that multi-path routing can achieve robustness, load balancing, congestion reduction, low power consumption, and higher throughput [MGLA96, PP03, GGSE01, YKT03].

This chapter provides an overview of the widely popular ICN architectures. The first sections of this chapter identify and elaborate the different features of an ICN architecture common to all the different architectures. The subsequent sections provide introductions to these different ICN architectures and how these features are handled in the specific architecture. A detailed description of Content Centric Networking (CCN) is presented as CCN was used as the basic architecture for the research done in this thesis work.

2.1 Information and Naming

The primary aspect of every ICN architecture is the focus on information or content. Unlike in current networking architectures where information is associated with the location (e.g., through a Universal Resource Locator (URL)), ICN focuses on the information itself and therefore, is indifferent to where the information resides. ICN names the information rather than the hosts on which they reside. This information, which we usually term as files or streams are broken down to segments similar to current networks and these segments are identified as chunks or objects. Each of these chunks has a unique name which a user can specify to access. Multiple copies of these chunks reside at multiple locations in an ICN based network. A user requesting a chunk may receive the copy of the chunk located nearest to the user [ADI+11].

2.2 Caching and Storage

Another key aspect of ICN is the use of caching to store content in the network. Nodes that have already seen a particular chunk may decide to hold a copy of that chunk.

Caching content requires a number of decisions that need to be made on *where*, *how* and *what* to cache. These architectures consider 2 alternative policies on where the caching is done [ADI+11]. The first alternative, called the *caching at the network edge* is where the edge nodes such as peers in P2P networks or through dedicated servers are made to hold replicated copies of content. The second alternative, called the *in-network caching* is where the caching of content is done under the control of the transport network, such as caching performed by the forwarding routers along the path of requested content.

Handling the *how* and *what* to cache is done using a *cache replacement policy* and a *cache decision policy*, respectively. The *cache decision policy* decides the type of content selected to cache. There are different policies such as Cache Always (ALWAYS) and Random with a Fixed Probability (FIX(P)). The *cache* *replacement policy* decides how new entries are inserted into the cache. There are a number of such policies such as Least Recently Used (LRU), First In First Out (FIFO) and Uniformly Distributed (UNIF) [RR11].

2.3 Routing

Routing in ICNs is based on the names associated with the content and not based on the name associated with the location as in current networks. Currently, there are 2 approaches being followed by the different ICN architectures, viz., *Name Resolution based Routing* and *Name based Routing* [ADI^{+}11].

Name Resolution based Routing refers to a 3 step approach where a name is resolved to a location specific to the underlying transport network, before retrieval of the content. This approach is usually overlaid on an existing transport network architecture such as TCP/IP. It is similar to the operation of a Domain Name System (DNS) in TCP/IP, but differs by not being limited to domain names (e.g., resolution of a set of predicates to a location). The steps of this approach are as follows.

1. Routing the content request message to a Name Resolution Service (NRS) to obtain an address or addresses of the location(s) where the content resides,

2. Routing the content request message to the location specified in Step 1,

3. Routing the data of the content to the requester.

The *Name based Routing* approach performs the retrieval of content by routing requests based on the name of the content until the request reaches a cache that holds a copy of the content. This approach has 2 steps.

1. Routing the content request message over the network where each ICN router forwards further based on the name,

2. Routing the data back to the requester following the reverse path of the path created during the propagation of the request.

This architecture follows a hop-by-hop message communication mechanism compared to the end-to-end mechanism adopted in *Name Resolution based Routing*.

2.4 Security

The content in ICN is secured by incorporating the security aspects in the content itself unlike current networks where the trust is associated with the location where the content resides. This form of security in ICN results in the ability for content to reside in any cache in the network, irrespective of whether the location is trusted or not [ADI^{+}11, DGOA10]. The following security characteristics are present in ICN by design.

- Content Integrity: Ability to identify accidental or intentional modifications to content, which is also referred to as *Self Certification* of content,
- Accountability: Ability for the content to be authenticated and associated to an entity such as a person or an organization,
- Confidentiality and Controlled Access: Ability to let only the eligible entities read the content,
- Availability: Ability to make content available and accessible for authorized entities.

2.5 ICN Architectures

There are number of ICN architectures that are being looked into by the research community. They have different approaches on how the fundamental features described above are addressed. In this section, an overview is provided to some of the most widely discussed ICN architectures, viz., Network of Information (NetInf), Publish-Subscribe Internet Routing Paradigm (PSIRP), Data Oriented Network Architecture (DONA) and Content Centric Networking (CCN).

These 4 architectures are described in brief in the following sub sections. CCN is revisited in a later section to provide detailed architectural and operational information.

2.5.1 Content Centric Networking (CCN)

The CCN architecture operates by routing the requests (Interests) for content to the content location through the routing infrastructure. When an Interest reaches a location that holds the requested content, which maybe the producer or a cache that has cached a copy of the content previously, this content will traverse back to the requester along the path on which the Interest arrived. CCN has a hop-by-hop transport mechanism and therefore, is able to be overlaid on different transport

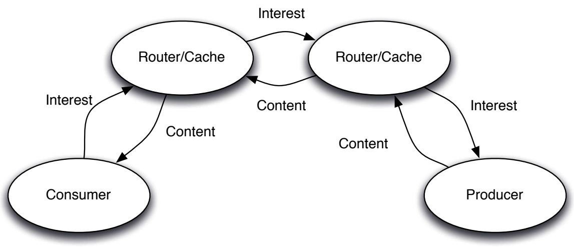


Figure 2.1: CCN Operations (Source: [ADI$^+$11])

network technologies such as TCP/IP or even over a Media Access Control (MAC) technology such as Ethernet [JST$^+$09]. Figure 2.1 shows the operations of CCN architecture.

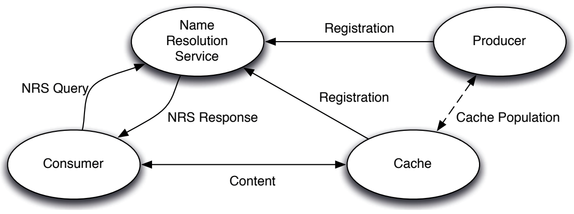


Figure 2.2: NetInf Operations (Source: [ADI$^+$11])

2.5.2 Network of Information (NetInf)

The operation of the NetInf architecture consists of a number of phases [Dan09]. The basic content retrieval process consist of the following 2 phases.

- Resolution Phase - A request for a content (which has a name or a set of predicates) is resolved to an address (i.e., location) of the underlying transport network, e.g., an Internet Protocol (IP) address. This is done through a Name Resolution Service (NRS).
- Retrieval Phase - Once the location is known, the retrieval of the content is done in this phase.

In addition to these phases, there are 2 other phases. The first one of these is where the NRS servers are populated with the content names and their locations in terms of the addresses of the underlying transport network. This is usually done by the content producer. The second phase (of other phases) is where caches are populated with the content. Figure 2.2 shows the operations of all these phases.

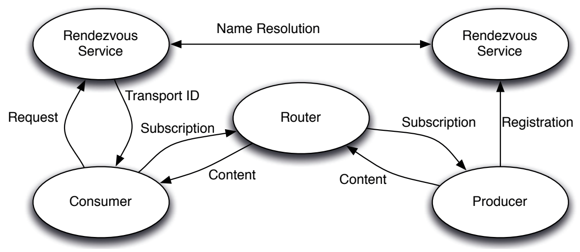


Figure 2.3: PSIRP Operations (Source: [ADI+11])

2.5.3 Publish-Subscribe Internet Routing Paradigm (PSIRP)

The PSIRP architecture works on the publish-subscribe principle [FTP11]. In this architecture, the creators of content publish them in the network and subscribers who are interested in the content retrieve them through subscriptions. A *rendezvous system* maps the published content to subscriptions and provides a Rendezvous Identifier (RI). The RI is an identifier for a communication channel. This identifier is then mapped to a *Transport ID* which is an identifier specific to the underlying transport network to retrieve the content. Figure 2.3 shows the operations of the PSIRP architecture.

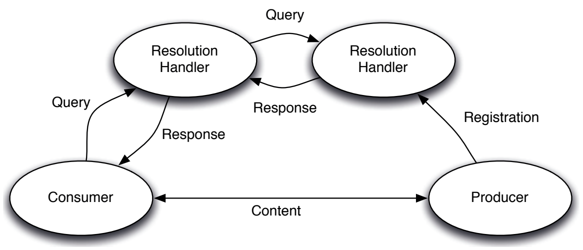
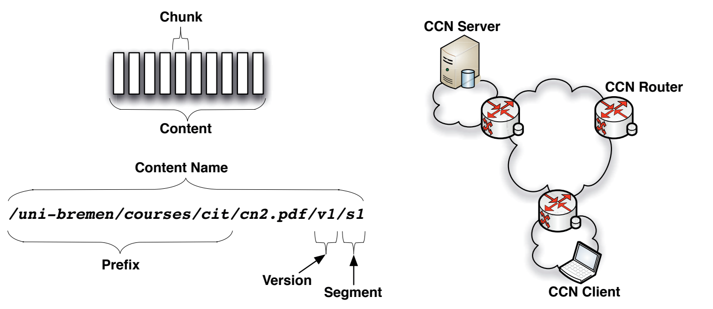


Figure 2.4: DONA Operations (Source: [ADI+11])

2.5.4 Data Oriented Network Architecture (DONA)

The DONA architecture operates by nodes registering content with the Resolution Handler (RH) nodes and requests for content being routed to these RH nodes [KCC+07]. The registrations have a lifetime and when the lifetime expires, they have to be renewed. The RH nodes have a hierarchical structure. Requests for content are routed by name using the hierarchical structure of DONA to discover content closer to the requester. Once the location of a content to retrieve is resolved, the underlying transport network such as TCP/IP is used to retrieve the content. Figure 2.4 shows the operations of DONA architecture.



(a) Content Naming and Structure (b) Types of Nodes in CCN

Figure 2.5: CCN Terms

2.6 Definitions of Terms

The architecture and the operations of CCN is described in [JST+09] using a number of terms. This section provides the definitions of these terms. Figure 2.5 shows a graphical view of the terms defined below.

- **Content**: A content is a complete file such as a video consisting of many bytes which a user wishes to retrieve from a provider in the network. Each content is identified by a name and a version. The name may have a hierarchical naming structure that may or may not follow the domain naming hierarchy of TCP/IP.
- **Chunk**: A content is split into multiple parts and each one of these parts is referred to as a chunk (also referred to as a **Segment**). A chunk is identified by a *Content Name* (given below).

- **Content Name**: A Content Name refers to the name uniquely identifying a Chunk, consisting of the name of the content to which the Chunk belongs, version identifier and segment identifier.
- **Prefix**: A Prefix identifies a part of the Content Name which is mainly used by the forwarding strategies.
- **CCN Node**: A CCN node is any computing device such as a user terminal that implements the CCN protocol stack for network communications.
- **CCN Client**: A type of CCN node that is deployed with a set of client applications (e.g., file download application) for a user to perform content retrievals (e.g., user terminal).
- **CCN Server**: A type of CCN node that is deployed with a set of applications that serves contents (e.g., a Web Server). It is differentiated from a CCN Router deployed with the caching functionality. A CCN Server hosts the contents it serves.
- **CCN Router**: A type of CCN node that is configured to perform 2 primary tasks, viz., forwarding and caching. A CCN Router performs the forwarding task by forwarding the requests for chunks it receives in the direction of where the chunks reside. When the requested chunks arrive, they are forwarded in the direction of the requesters. The caching task is performed by storing the chunks that traverses the CCN Router. Before performing the forwarding of a request, a CCN Router checks its cache to see the possibility of serving the requests.
- **Interest Message**: The CCN message that carries the Content Name of a chunk being requested (described in detail in Section 2.7.1).
- **Content Object Message**: The CCN message that carries the Content Name and the data of the chunk that was requested by a preceding Interest message (described in detail in Section 2.7.1).

2.7 Content Centric Networking (CCN)

The CCN architecture was proposed by Palo Alto Research Center (PARC) in a seminal publication published in 2009 [JST$^+$09]. This architecture, which was revolutionary at the time, had at its core, the concept of enabling the retrieval of content using the name of the content rather than the name of the location of the content. Different from other content-aware infrastructures such as NetInf and CDN, which are mostly deployed in the application layer, CCN changes the universally accepted common network protocol layers from TCP/IP to named content. The protocol stack of CCN still keeps the same hourglass-shaped architecture of TCP/IP, allowing lower and upper layer networking technologies to innovate without unnecessary constraints. However, the "thin waist" of TCP/IP, viz., the IP layer, is replaced by chunks of named content (Figure 2.6).

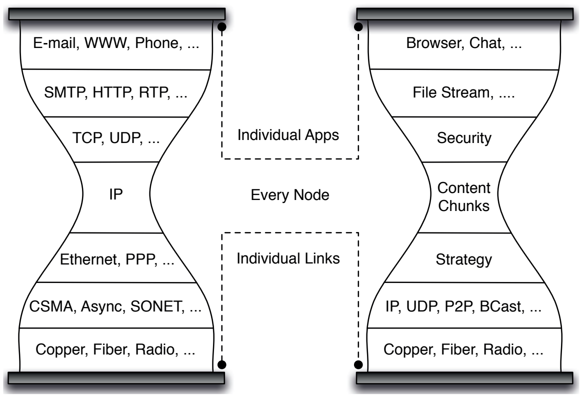


Figure 2.6: Protocol Stack Comparison of TCP/IP and CCN - Change of the "Thin Waist" from IP packets to named Content Chunks (Source: [JST^{+}09])

The protocol stack architecture of CCN results in significant benefits.

- This architecture frees up the network from middleware that is used for mapping the relationships between the application's content-centered model and the Internet's address-centered model. This results in simplified network implementations and configurations which in turn results in increased communication efficiencies.
- Due to the simpler relationship that the CCN specific layers of the protocol stack has with the data link layers, maximum advantage can be availed through multiple simultaneous connectivity options (e.g., Ethernet, Long Term Evolution (LTE), Bluetooth and Wireless Local Area Networking (WLAN)) [JST^{+}09]
- CCN provides a much higher level of security by securing the content itself instead of the communication path as in TCP/IP.

The CCN related layers of the protocol stack of CCN introduces a number of elements for its operation. The following sections describe these elements and the operations.

2.7.1 CCN Node Architecture

Communications in CCN are triggered by user requests for content. All communications in CCN are handled using two message types, viz., *Interest* and *Content Object* (Figure 2.7). When users request content, they broadcast Interest messages specifying (at a minimum) the name of the required content over one or more *Faces*. A *Face* is an interface to either the network (i.e., network attachment) or to an application being executed on a CCN node. Any CCN node that receives this Interest and holds a copy of the requested chunk, may reply with a Content Object message containing the requested content.

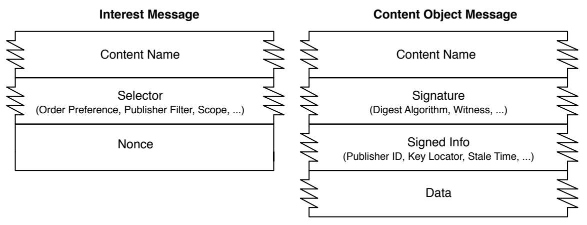

Interest Message
Content Name
Selector
(Order Preference, Publisher Filter, Scope, ...)
Nonce
Content Object Message
Content Name
Signature
(Digest Algorithm, Witness, ...)
Signed Info
(Publisher ID, Key Locator, Stale Time, ...)
Data

Figure 2.7: CCN Message Formats (Source: [JST$^+$09])

The Interest message consist of the fields *Content Name*, *Selector* and *Nonce*. The Content Name refers to the name of the Chunk being requested, that includes the name of the content, the version and the segment of the Chunk. The Selector is a means by which the originator of the message uses to provide additional information to be used by the forwarding CCN Routers when making forwarding decisions. The Nonce is generated by the originator of the message and is used to prevent the build up of path loops during the propagation of the Interest.

The Content Object message consist of the fields *Content Name*, *Signature*, *Signed Info* and *Data*. The Content Name is the same as the Content Name in the Interest message. The Data field carries the payload of the chunked data, i.e., the data associated with the Chunk. Every Content Object message is signed by the creator of the Chunk. The Signature and the Signed Info fields carry the signature and the information associated with the signer, respectively.

The core operations of a CCN node is handled by the forwarding engine which manages its operations through three data structures, viz., Forwarding Information

Base (FIB), Content Store (CS) and Pending Interest Table (PIT). The term CCN node may refer either to a node that resides in a CCN network that acts as a router or to an end-node that usually produces or consumes content. The request for contents by a user is initiated by generating an *Interest* by an application. In CCN, a content is segmented into what are called *Chunks*. Each of these *Chunks* have a name that includes a sequence number and a version number. The fundamental operation of CCN is as follows.

The arrival of an Interest message results in the following actions.

1. The name specified in the Interest is searched in the CS for availability of the requested Chunk. If the Chunk is available, a Content Object is created and is sent over the Interest arrival Face. The Interest is then discarded.

2. If the Chunk is not available in the CS, the PIT is checked to see if an Interest has been received with the same Chunk name before. An entry in the PIT indicates the receipt of an Interest with the same name previously. If such an Interest was not seen before, a new PIT entry is created with the Chunk name together with the arrival Face of the Interest.

3. Once the PIT entry is created with the information in the Interest (i.e., a "bread crumb" is dropped), this Interest is further forwarded in the direction of the content. To identify the Face or Faces through which the Interest is to be forwarded, the FIB is searched for the Chunk name. The search is done using the Longest Prefix Matching (LPM) algorithm and together with the forwarding strategy employed in the CCN node, the Interest is forwarded to the identified Faces. The Interest is discarded subsequently.

4. When a PIT entry is available, the Face through which the Interest arrived is added to the PIT entry and the Interest is discarded.

5. Every PIT entry has a lifetime and when the lifetime is reached, they are removed from the PIT.

The arrival of a Content Object results in the following actions.

1. The name specified in the Content Object is searched in the PIT to identify whether this Content Object is being received due to a previously propagated Interest.

2. If a PIT entry exists, the received Content Object is sent over the Faces listed in the PIT entry and the PIT entry is removed (i.e., "bread crumb" is removed).

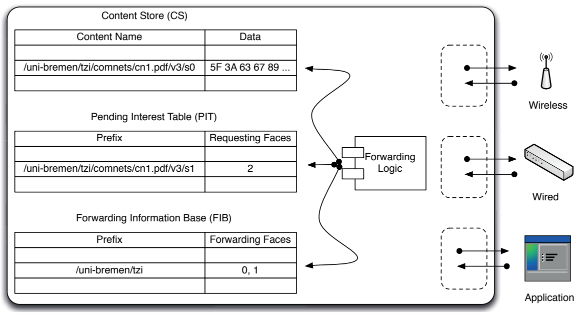


Figure 2.8: CCN Node Architecture (Source: [JST+09])

3. If a corresponding PIT entry does not exist, then the Content Object is disregarded.
4. If the CCN node is configured as a cache and decides to store the Content Object based on the employed caching policies, the Chunk is inserted into the CS.

2.7.2 Flow Balance

TCP/IP based networks employ Flow and Congestion Control (F&CC) mechanisms to adapt a communication session for packet losses or congestion in the networks to maintain a flow balance. The F&CC in these networks occur at the end points of a communication session based on the information available at these end points. One of the main problems of end point based flow and congestion control is the slower reaction to changes in the network conditions. In contrast to TCP/IP, CCN F&CC occurs at each hop along a path to where the contents reside [JST+09]. CCN has two fundamental operational characteristics that maintain the flow balance in communications.

- One Interest in CCN retrieves only one Content Object (i.e., Chunk) at most. Furthermore, the PIT at each CCN router, forwards only the first Interest received and a subsequent Interest for the same Chunk is only updated in the PIT with the arrival Face.
- Content Objects must arrive only at locations where there was a corresponding Interest that went through, i.e., if a Content Object is unable to find a matching PIT entry at a CCN router, it will be discarded.

These are the basic rules that ensure a balance in the many flows that are present in CCN networks. In short, CCN guarantees that at most only one copy of any piece of Content Object travels over any link, which efficiently reduces the traffic load in networks.

2.7.3 Loop-free Forwarding

An additional advantage of the flow balancing mechanism is the prevention of the formation of loops with Interest forwarding at CCN routers. Every Interest in CCN carries a random nonce value with it which is used by CCN routers when making forwarding decisions. When a router detects a duplicate nonce value, that Interest is discarded. Thereby, the forwarding mechanism ensures that the same Interest is prevented from being forwarded over and over again. This process prevents the formation of loops in Interest forwarding and this in turn will prevent loops forming for Content Objects as the Content Objects always follow the trail (i.e., the "bread crumbs") left by the propagating Interests. This loop-free forwarding mechanism allows CCN to take maximum advantage of using multiple simultaneous attachments to networks (e.g., Ethernet, WLAN).

2.7.4 Forwarding Strategies

The original architecture of CCN proposes two forwarding strategies [JST$^+$09] . The first strategy, identified in this work as the *standard strategy* employs flooding of Interests over the potential Faces to reach the content. The second strategy, identified as the *best-face strategy* uses a single Face to forward the Interests.

2.7.4.1 *Standard Strategy*

The flooding strategy (*standard strategy*) sends the same Interest over all *BroadcastCapable* faces and if there is no response, all the other *Faces* are tried sequentially. This strategy guarantees the retrieval of content from the closest CCN cache which holds a copy of the requested Chunk. However, flooding imposes a heavy burden on the network due to the high volume of network traffic.

2.7.4.2 *Best-face Strategy*

The *best-face strategy* selects a face to forward Interests based on information collected about communications. In CCN, a node is guaranteed to receive Content Objects sent by other CCN nodes in response to previously generated Interests. Therefore, each CCN node can gain "*fine grained*, *per-prefix*, *per-face*" [JST$^+$09] performance information to adaptively choose one or more Faces from the list of entries in the FIB. The Strategy Layer of the CCN protocol stack (Figure 2.6) is tasked with collecting information, by occasionally broadcasting an Interest as a probe to all Faces associated with the prefix. Based on the collected information, the Strategy Layer chooses one best Face for subsequent Interests with the same prefix. The best Face is defined in [JST$^+$09] as the Face through which the first returning Content Object is received in response to a previously sent probe Interest. All of the paths selected using the *best-face* strategy will be the shortest paths in terms of delay, to the content provider or the CCN cache. In comparison to the *standard strategy*, the *best-face* ensures that Content Objects arrive from the nearest cache in most cases which additionally results in reduced unnecessary traffic in the network.

Even though the *best-face* strategy results in the creation of a shortest path, shortest path routing has been proven to cause unbalanced traffic distributions, since links on frequently used shortest paths become increasingly congested, while other links are underutilized [Vil99]. Additionally, use of single paths have the issue of not being able to satisfy service level agreements on bandwidth requirements of users and applications.

2.7.5 Distributed Caching

One of the fundamental properties of CCN is the functionality which allows a CCN node to cache the contents traversing through it. With caching, a user's requested content may be found in a CCN node in the vicinity of the user instead of the requests traveling all the way to the content provider, as it currently happens in today's networks. Figure 2.9 provides a graphical view of conventional and distributed caching. Caching of content results in lower delays for content retrievals for the user. Furthermore, since the requests for content are served from caches near the user, the burden on the networks are reduced. This results in benefits for network operators and is equally beneficial for other users due to reduced network traffic.

Distributed caching naturally favors multi-point to multi-point communications due to the same content being available at a number of caches. Therefore, users have the possibility of utilizing diverse paths to retrieve the same content, possibly in parts. Diversity is considered as one of the main criteria of selecting multiple paths.

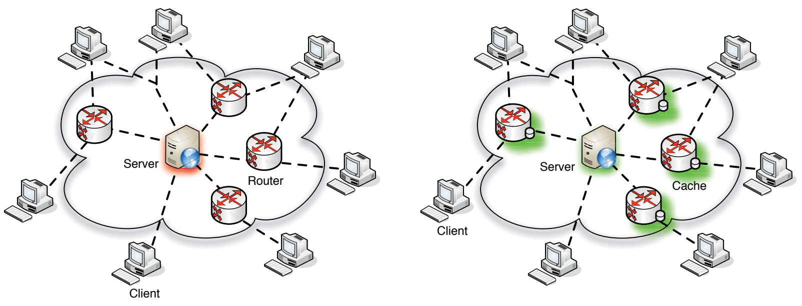


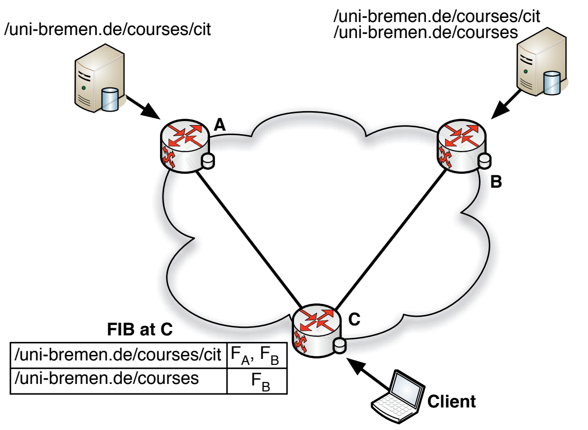
Figure 2.9: Conventional Content Retrievals (left) and Content Retrievals with Distributed Caching (right)

2.7.6 Routing

Routing in CCN refers to the process of forwarding Interests received at a CCN node. The basis for all routing activities in CCN is the FIB. The FIB has a similar purpose as the routing table in IP but unlike IP routing tables, a FIB entry points to one or more Faces which are used to forward Interests. Any routing scheme that works well in IP is considered to equally work well for CCN due to the CCN forwarding model being a strict superset of the IP model with the same semantics relevant to routing (hierarchical name aggregation with longest-prefix match lookup) but without the restrictions on multi-source and multi-destination routing [JST$^+$09]. When considering the overall routing process, CCN, similar to IP follows the same two phase approach to routing Interests that arrive at a node.

- The FIB population phase or the topology sensing phase is where a FIB is filled with entries pointing to locations where the content resides. As stated in [JST$^+$09], the process of populating a FIB could be done using an unmodified link-state based Interior Gateway Protocol (IGP) type protocol such as the Intermediate System to Intermediate System (IS-IS) routing protocol or the Open Shortest Path First (OSPF) routing protocol.

- The forwarding phase is the process of forwarding Interests to one or more Faces based on the adopted forwarding strategy (Section 2.7.4). Generally, all the forwarding strategies perform the forwarding to the Faces associated with a prefix entry in the FIB after performing a longest-prefix match on the FIB.



/uni-bremen.de/courses/cit	F_A, F_B
/uni-bremen.de/courses	F_B

Figure 2.10: Populating the FIB and Forwarding in CCN

Figure 2.10 shows a CCN based network with 3 CCN routers to perform forwarding. The router *A* hears an announcement from a neighbor about the availability of content with the prefix "*/uni-bremen.de/courses/cit*". Router *A* packages this information in an IGP Link-State Advertisement (LSA) message and floods it to the other CCN nodes in the vicinity. When *C* hears this message, it creates an entry in its FIB with prefix and the Face through which it was received (i.e., F_B). Subsequently, another router, *B* announces through an IGP LSA message the availability of content for two prefixes, viz., "*/uni-bremen.de/courses*" and "*/uni-bremen.de/courses/cit*". These messages are also heard by *C* and *C* updates its FIB accordingly. The aforementioned activities are the activities of the FIB population phase. A client connected to *C* requests the content for "*/uni-bremen.de/courses/cit/cn2.pdf*" through Interests. Based on the FIB, it forwards them to both *A* and *B*. These routers in turn forward them to the repositories. This part of the activities belong to the forwarding phase.

Though the explanations followed a sequential procedure, in reality the FIB population phase may also occur after the forwarding phase due to new content being available and also due to relocation of content repositories.

Furthermore, the LSA messages used above are neither Interest nor Content Object messages. They are simply used by a CCN router to inform and be informed about the prefixes served by oneself and other CCN routers in the network.

2.7.7 CCN and TCP/IP

TCP/IP is the most widely used networking architecture in the world. This section describes the relationship between TCP/IP and CCN. This relationship has two facets, viz., the architectural and co-existent relationships.

2.7.7.1 Architectural Relationship

As described above, CCN follows a similar network architecture as TCP/IP. The protocol stack of CCN adopts the TCP/IP "thin waist" layering of protocols. The *Content Chunks* layer, the "thin waist" of CCN (Figure 2.6) resembles the same simplicity that the IP layer has in TCP/IP. The Interest and Content Object messages, similar to IP packets carry information that enables a simple forwarding process based on the name of the content. The forwarding process uses the FIB similar to TCP/IP using a routing table, to perform a destination look-up based on LPM. The main differences are the use of names in CCN to perform the forwarding unlike the use of host addresses in TCP/IP and the possibility of LPM returning multiple destinations to use for forwarding in CCN.

By adopting the "thin waist" at the *Content Chunks* layer, CCN, similar to TCP/IP, allows the innovations to occur above and below the *Content Chunks* layer without affecting the fundamental hop-by-hop forwarding mechanism. Though both architectures adopt the hop-by-hop forwarding of messages, TCP/IP has an end-to-end focus in communications while CCN is unaware of the nodes beyond the next hop. This is due to the use of caching in CCN where a CCN node is unaware of the source of contents while the end points in a TCP/IP communication session stays constant during communications.

2.7.7.2 Co-existence

CCN is an independent communication architecture that is able to be layered over any underlying link technology. For example, the CCN protocol stack is able to operate over Ethernet. CCN simply requires a means of communicating with a connected next hop to send and receive Interests and Content Objects.

But, the evolutionary nature of progress in networking, especially with the Internet requires any new technology to operate over TCP/IP and equally, side-by-side with the existing applications. CCN is able to operate over TCP/IP using any of the transport protocols to communicate with the next hops.

The evaluations performed in this thesis uses the evolutionary approach where the CCN model is layered over User Datagram Protocol (UDP) communications.

2.7.8 CCN and Application Layer Caching Solutions

CDNs, P2P overlays and Hypertext Transfer Protocol (HTTP) Proxies are application layer solutions that enable caching of contents for the benefit of requesting users. These solutions are extensions to the application layer of the TCP/IP architecture where special purpose servers cache contents and the networking environment is modified to make the requests for contents arrive at these servers. Due to these solutions being application layer specific, the cached content too is specific to the the type of contents handled by those specific applications (e.g., CDNs cache HTTP web content).

On the other hand, with CCN, the architecture has built in support to cache content at the *Content Chunk* layer and is therefore not restricted to a specific set of nodes dedicated to perform the task of caching content. Moreover, caches in CCN cache any type of content irrespective of the type of application that generates these contents.

3 Multi-path Forwarding in CCN

Most computing devices of today are equipped with multiple networking technologies and hence, have the ability to use multiple paths over different networks. The objective of this thesis is to investigate the use of multi-path in Content Centric Networking (CCN). The widest possible experience in using multiple paths for communications is mostly available in Internet Protocol (TCP/IP) based networks. Research done shows that multi-path routing can achieve robustness, load balancing, congestion reduction, low power consumption, and higher throughput [MGLA96, PP03, GGSE01, YKT03]. But, when considering multi-path with CCN, however, research is limited to a few work [YAM$^+$13, YAW$^+$12, RR11] which were published during the period of this thesis. [RR11] uses replication of Interests to multi-path to evaluate the performance of caching while [YAM$^+$13, YAW$^+$12] use multi-path as backup paths to improve reliability. The work presented in this thesis goes beyond to focus on a number of additional mechanisms to better use multiple paths in CCN.

This chapter describes the design of an effective and reliable On-demand Multi-path - Interest Forwarding (OMP-IF) strategy to discover and use multiple paths simultaneously. Once the paths are discovered, Interests are distributed based on path delays to avoid flooding and to guarantee efficient and robust data delivery. The OMP-IF strategy makes decisions dynamically on a hop-by-hop basis, making transmission more efficient. The chapter starts by detailing a number of aspects related to multi-path in general, subsequently moving on to describing the multi-path forwarding strategy and the related mechanisms proposed in this thesis. The work done on the OMP-IF strategy has been described in [UZKG14].

3.1 Feasibility of Multi-path in CCN

Compared to the multi-path routing strategies proposed for TCP/IP networks, CCN inherently has a number of capabilities that allows it to take advantage of multiple paths between content requesters and providers [FRHB13, MD01, GDS$^+$03, MBNP06]. Firstly, distributed in-network caching in CCN naturally favors multi-point to multi-point communications, allowing every CCN router to cache the data flowing through it. Therefore, Interests do not have to be routed to one single source, since several nodes in a network can provide the same content (Figure 2.9). Secondly, CCN routers do not simply forward Interests but also involve themselves in forwarding decisions thereby possessing the ability to consider different factors (e.g., path performance as in [YAW+12]). Thirdly, TCP/IP is forced to construct loop-free forwarding topologies unlike CCN where a *nonce* in Interests takes care of this.

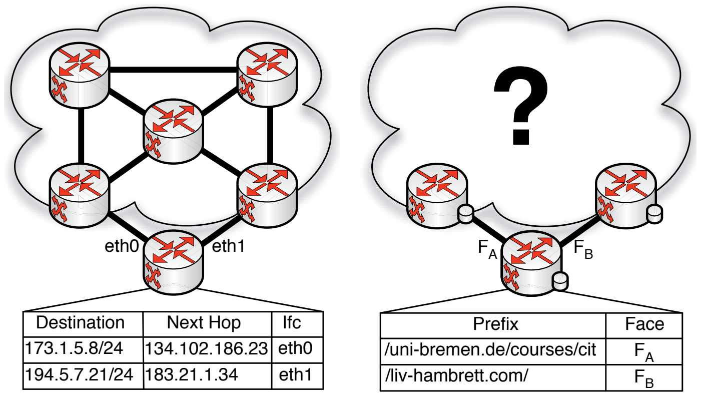
3.1.1 CCN Does Not Require Explicit Discovery of Shortest Paths

It is obvious that name based routing and the forwarding architecture in CCN has a huge difference with the address-based TCP/IP networking. But, they also share some common characteristics. CCN could benefit from some notions of Internet Protocol (IP) routing protocols. As stated in Section 2.7.6, the creators of CCN argue that CCN can populate a Forwarding Information Base (FIB) in the same manner as the Link State Routing (LSR) protocols do in TCP/IP networks. TCP/IP forwarding and CCN forwarding are almost identical since they both use prefix-based longest prefix match lookups to find local neighbor(s) "closer" to the identifier matched. However, in TCP/IP, the prefix is the IP address of a host or a network while in CCN, it is an alpha-numeric name. Further, with TCP/IP, the neighbor is indicated by the next hop IP address while in CCN a list of one or more Faces are given.

In TCP/IP networks, the route discovery process through LSR is performed in two stages; *distribution of maps* and *calculation of routing table*. The task of the first stage is to provide a map of the network to every node by flooding link state information throughout a network of routers. By aggregating the flooded network information, each router can independently build a database of the network's topology. In the second stage, the routing tables are generated by inspecting the maps. Each node independently calculates the shortest path from itself to every other node in the network by running an algorithm (a variant of Dijkstra's algorithm) over the map. Once a shortest path is identified, the information of each node on the shortest path is inserted as the corresponding routing table entry.

When using LSR protocols in TCP/IP networks, a node announces Internet Protocol version 4 (IPv4) or Internet Protocol version 6 (IPv6) prefixes via LSA. An IP address acts as the identity of a host, hence the LSA shows an exact connectivity between each node. Nodes build up an explicit map of the network according to the collected link-state information (Figure 3.1(a)).

In contrast to TCP/IP, every node in CCN announces the prefixes of contents which the node can serve. These prefixes name the content and does not allude to



Destination	Next Hop	Ifc
173.1.5.8/24	134.102.186.23	eth0
194.5.7.21/24	183.21.1.34	eth1

Prefix	Face
/uni-bremen.de/courses/cit	F_A
/liv-hambrett.com/	F_B

(a) In TCP/IP networks, every node can learn the topology of the network through LSA protocols

(b) In CCN, every node only learn the available Content Prefixes attached to every *Face* after populating the FIB

Figure 3.1: LSA Protocol in TCP/IP vs. FIB Population in CCN

any identities of the nodes that hold the content. Therefore, each CCN router can only fill the FIB table with the Faces through which the content associated with the prefixes are accessible and further, these prefixes are not possible to be used to build a map of the network (Figure 3.1(b)).

In addition to the LSR based solution above, the authors of [JST$^+$09] declare that it is also possible to overlay CCN on existing Intermediate System to Intermediate System (IS-IS) or Open Shortest Path First (OSPF) networks, without any modifications to the network or the routers. As described by [Col98, VSA07], IS-IS and OSPF can specify directly connected resources through a Type-Length-Value (TLV) scheme using the Media Access Control (MAC) address to identify a neighbor. Using this mechanism, CCN routers can learn about the underlying network topology, announce their position in that topology by synchronizing the database with TCP/IP routers and flood their serving prefixes through prefix announcements using a CCN specific TLV. However, in a pure CCN based network which is not overlaid on TCP/IP, a custom FIB population protocol is required that may still use the MAC addresses to identify nodes.

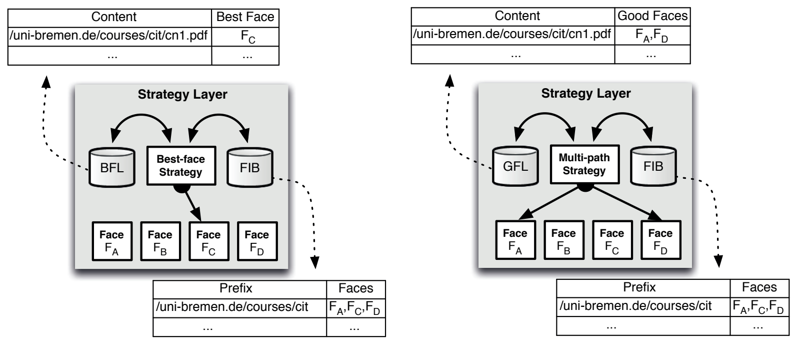
In spite of the above mentioned solutions, CCN however, does not require to perform a shortest path computation as CCN benefits from *"inexplicit routing"* as stated in [JST$^+$09]. Through *inexplicit routing*, CCN is able to automatically identify close to optimal paths in terms of bandwidth and delay due to the operation of CCN. In TCP/IP networks, multiple announcements of the same prefix results in only one node being selected to forward matching traffic while in a CCN based network, a node may forward matching Interests to all the announcers. This is due to the differences in the operational semantics of CCN and TCP/IP where a prefix announcement from an Interior Gateway Protocol (IGP) router in a TCP/IP network implies that *"all the hosts with this prefix can be reached via me"* while in a CCN network, such an announcement implies that *"some of the content with this prefix can be reached via me"* [JST+09].

A prefix announcement in CCN does not automatically imply that all content covered by an announced prefix is adjacent to the receiving CCN router. Since multiple announcements for the same prefix are received, what is possible is to forward an Interest to all the CCN nodes that announced the prefix, without worrying about loops being formed (as in TCP/IP). But, forwarding to all faces results in flooding networks with Interests which may result in degradation of the overall performance. This flooding strategy does not have to be executed for every Interest being forwarded as the operation of the *Strategy Layer* of the CCN protocol stack (Section 2.7) can be extended, as envisioned by the creators of CCN to control the flow of Interests through a node. In fact, the creators of CCN themselves have identified a forwarding strategy called the *Best-face* strategy to avoid the issues with flooding where a single Face is used to forward Interests.

The work presented in this thesis, therefore, focusses on developing mechanisms to perform multi-path content retrievals by extending the operation of the *Strategy Layer*.

3.1.2 Strategy Layer

One of the extensible protocol layers of CCN is the *Strategy Layer* (Section 2.7). This layer is similar in functionality to the *Network Layer* in the Open Systems Interconnection (OSI) model or the *Internet Layer* of TCP/IP, but with the possibility of incorporating extensions that go beyond the simple forwarding mechanism (i.e., routing) based on longest prefix matching. In CCN, this functionality is collectively referred to as a *Forwarding Strategy*. The *best-face Strategy* proposed by [JST+09], determines the shortest path to use to forward Interests to retrieve the content. The shortest path (i.e., Face connected to the shortest path) is identified by occasionally executing broadcast experiments to identify a Face as the *best-face* (Figure 3.2(a)). During the experiment, the Face through which the first requested Content Object arrives is considered as the *best-face*. The Best Face List (BFL) is used to hold a list of best *Faces* associated with each of the currently requested content.



(a) Periodic broadcasting of Interests to Determine the *best-face*

(b) Multiple *Faces* are Selected as *Good Faces* to Retrieve Content

Figure 3.2: Operation of Strategy Layer

The work presented in this thesis, extends the mechanisms described in [JST$^+$09] for the *best-face Strategy* to perform multi-path content retrievals. Instead of the BFL, a Good Face List (GFL) is introduced to hold a number of *Good Faces* for each content being downloaded as shown in Figure 3.2(b).

The following sections describe the *Multi-path Forwarding Strategy* proposed, in detail.

3.1.3 Processes Involved in Multi-path

The implementation of multi-path involves the processes of establishment, use and maintenance of the multiple paths for CCN.

3.1.3.1 Discovery Process

The discovery process refers to the establishment of multiple paths to retrieve content. The discovery involves the use of a mechanism that is guided by a set of criteria and the establishment of such multiple paths may occur at a CCN node or a CCN router. Two of the key questions in discovering multiple paths is; *"how should the paths be selected"* and *"how many paths are sufficient"*.

[MBNP06] and [NZ01] states that performance improvements with the use of multiple paths in TCP/IP based networks is not proportionately correlated to the number of paths and the best is to use a few paths (such as two or three). There are a couple of justifications for using a few paths.

- More paths result in more overheads to establish and maintain multiple paths.
- Considerable increase of the complexity of the path usage algorithm to distribute traffic to the multiple paths [NZ01].

There are a number of criteria that are used to select multiple paths. The most commonly identified criterion is the diversity of paths, i.e., the minimality of influence that one path has over the others. Creation of disjoint multiple paths is a means of achieving diversity between paths. Disjoint paths have the advantages of resource aggregation and higher fault tolerance. Failures in non-disjoint path elements (nodes and links), may result in failures in all discovered paths.

In addition to diversity, other criteria used in path selection also include hop count, delay, available bandwidth, Signal-to-Noise Ratio (SNR), etc. as described in [NL07, HALA07, CBR04, LLP^{+}01, MAK^{+}08] with research done in TCP/IP based networks.

3.1.3.2 Utilization Process

Once the multiple paths are discovered, they should be used in a manner that achieves the objectives of resource aggregation, fault tolerance, etc. In a broader sense, there are two possible ways of using discovered paths as stated in [Kul09]. The first is the use of multiple paths, one at a time to improve network performance which is called Alternate Path Routing (APR) and also referred to as backup path routing. The second is where the multiple paths are used simultaneously by distributing traffic to the paths to aggregate the available bandwidth or to replicate the traffic to the paths to improve reliability and this is called Simultaneous Use of Multi-path (SUM) routing.

3.1.3.3 Maintenance Process

Maintenance of the paths is an important aspect of using multiple paths considering the dynamic nature of networks. Paths may be unavailable due to node relocations or link breaks along the created paths. This may occur in stationary networks as well as in mobile networks.

In the case of APR, where the alternate paths are unused until required, there must be mechanisms to keep the paths alive and to obtain continuous information on the availability of paths. With SUM routing on the other hand, the process of checking the validity of the created paths does not require an extra effort since those paths are continuously being used.

Once the knowledge of a path breakage is received, a decision has to be made on whether a re-discovery is required. The knowledge of a path breakage may also be due to a preemptive mechanism that predicts a future link break as in [TUG^{+}08]. The suitability of the existing paths and the overheads (e.g., disruptions) that may be incurred due to a re-discovery have to be taken into account when making this decision.

3.1.4 Nature of Multi-path in CCN

Research done in TCP/IP networks related to multi-path always considers the discovery of multiple end-to-end paths after identifying the best next hop to use for each path [Kul09, FRHB13]. From the analysis of Section 3.1.1, it is seen that an exact map of the network cannot be built in CCN to identify a list of the shortest paths to the required content. Similarly, a pure source routing mechanism [Pos81] is also not possible to be adopted due to CCN being a name-based networking environment [JST^{+}09]. An observation that could be made is that the operation of the propagation of Interest messages and the return of the corresponding *Content* messages are similar to the control messaging in on-demand routing protocols such as Ad hoc On-Demand Distance Vector (AODV) routing protocol [PBRD03]. Therefore, it can be said that the communication between a content consumer (i.e., a client) and the content location (i.e., content producer or a cache) is exploration rather than point-to-point based.

During this exploration process, each node along the path makes local decisions using the *Forwarding Strategy* (Section 3.1.2) to propagate Interests further. Therefore, discovering a path in CCN is a *Hop-by-hop On-demand* process and the main focus of this thesis is the identification of a *Hop-by-hop On-demand Multi-path Forwarding Strategy* and its related mechanisms.

3.2 On-demand Multi-path - Interest Forwarding (OMP-IF) Strategy

The OMP-IF strategy is a forwarding strategy that operates at the *Strategy Layer* of the CCN protocol stack to enable multi-path content retrievals. It is designed to take advantage of the built-in properties of CCN to perform multi-path. The OMP-IF strategy identifies a set of node-disjoint paths to content locations and uses these paths based on the path characteristics.

3.2.1 Design Goals

Neither the routers nor the end-points in CCN can determine or obtain knowledge of the whole path with local information. Therefore, the identification of multiple paths is considered as a selection of a set of *good faces* at each node (Figure 3.2(b)). The forwarding strategy that identifies, uses and maintains these *good faces* (i.e., OMP-IF Strategy) is based on,

- **Path Selection**: The identification of the multiple paths by CCN nodes is based on the paths being *node disjoint* from each other. Node disjoint refers to the creation of paths that do not have common nodes along two or more paths. Path diversity is a frequently considered criterion for paths selected due to the ability to aggregate bandwidth. Further, when considering dynamic network topologies, especially with mobility of nodes, disjoint paths have a higher fault tolerance due to the failure of one node or link along a path not affecting the performance of the other paths.

- **Hop-by-Hop**: The path discovery process is assisted by the nodes along the different paths making local decisions based on local information. Therefore, the creation of the paths is a result of the actions taken hop-by-hop until the location of content is reached.

- **Splitting Strategy**: A CCN content (or file) consists of a number of Chunks. The splitting strategy distributes the Interests of each content to the discovered multiple paths. Due to the use of the splitting strategy, the Content Objects would also arrive along the same path in which the corresponding Interest travelled, thereby being able to avail the benefits of bandwidth aggregation. An alternative to the splitting strategy is the *Replication Strategy* where a copy of the same Interests is sent over the multiple paths. But, this strategy is not considered due to its ineffectiveness as shown in [RR11].

- **Number of Paths Used**: A CCN node may have multiple *Faces* connected to different networks and applications. As described in Section 3.1.3.1, most benefits of multi-path is achievable through the use of a few paths. Therefore, the usage of multiple paths is limited to the two initially discovered paths, though the discovery process itself may discover more than two paths.

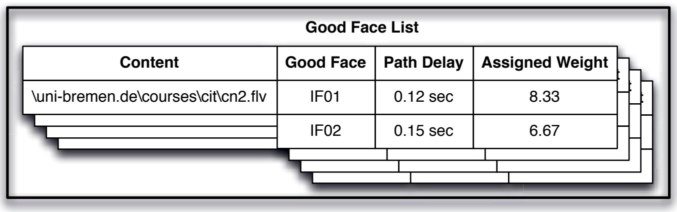
- **Path Quality**: The quality information associated with the used paths are collected and analyzed in real-time through the communications being done over these paths. A threshold based mechanism is used to determine the usability of the paths and when the quality of a path being used goes below the threshold, a backup path is used from the discovered paths. If no paths are available, a path re-discovery is triggered.

- **No Messaging Overheads**: One of the most simplistic architectural features of CCN is the usage of only two message types to request and receive content, i.e., Interest and Content Object messages respectively. The path discover, use and maintenance is done using these two messages without introducing any new message types. Additionally, the existing format of these messages are used without introducing any additional fields.

3.2.2 Strategy Layer Operations and Good Face List (GFL)

The OMP-IF strategy that operates at the Strategy Layer (SL) of the CCN protocol stack, relies on the forwarding information in the FIB. The FIB may contain a number of Faces for a given prefix and the OMP-IF strategy identifies several faces out of these, listing them as *good faces*. The population of the FIB may occur as explained in Section 2.7.6.

A data structure named Good Face List (GFL) stores the selected faces for currently active content downloads. Figure 3.3 shows an example of the GFL. Every GFL entry contains the name of the content for which a path discovery was done or initiated and a list of Faces that were identified as being the *good faces* for the corresponding content. Unlike the entries in the FIB, the entries in the GFL are created on-demand. When a node requires to forward an Interest, a search is done in the GFL with the name of the content, i.e., Content Name without the segment identifier (see Section 2.6). If an exact match is found in the GFL, the Interest is forwarded to one of the Faces listed as *good faces*. The Face that is picked from the list is decided based on the distribution mechanism used (Section 3.2.4).


Good Face List

Content	Good Face	Path Delay	Assigned Weight
\uni-bremen.de\courses\cit\cn2.flv	IF01	0.12 sec	8.33
	IF02	0.15 sec	6.67

Figure 3.3: Structure of GFL

When no entry in the GFL is found, the Interest is forwarded to all the Faces listed in the FIB (i.e., replicated) and an empty entry is inserted in the GFL. The *good faces* for this entry gets updated when a Content Object is received over the Faces. The entries in the GFL is timed out and finally removed when unused to prevent unnecessary memory usage and to minimize long GFL lookup times. Figure 3.4 shows how the OMP-IF strategy creates the GFL and uses it to forward Interests. The *Underlay Layer* refers to the protocol layer over which CCN is overlaid. An example is TCP/IP.

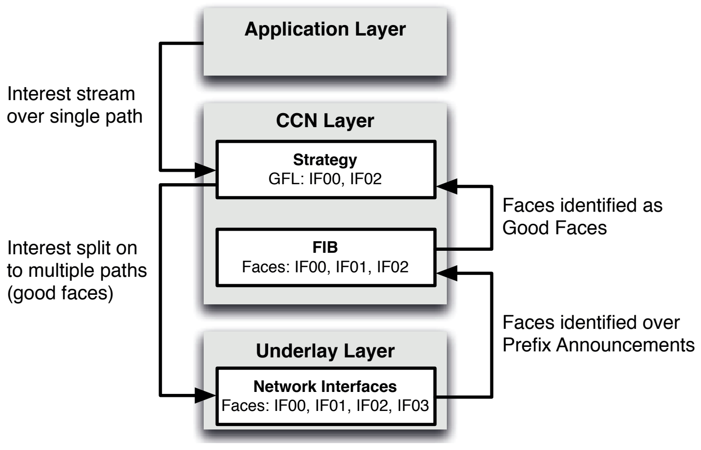

Application Layer
Interest stream over single path
CCN Layer
Strategy
GFL: IF00, IF02
Faces identified as Good Faces
Interest split on to multiple paths (good faces)
FIB
Faces: IF00, IF01, IF02
Faces identified over Prefix Announcements
Underlay Layer
Network Interfaces
Faces: IF00, IF01, IF02, IF03

Figure 3.4: Selection of *good faces*

Figure 3.5 shows the different states and the transitions of a GFL entry. *Idle* refers to the state where no GFL entry exists and when the first Interest arrives, the path discovery process commences. During *Discovery*, the Interests are replicated to all the faces identified in the applicable FIB entry. When the Content Objects arrives, the paths are made. As soon as at least one path is made, the subsequent Interest are forwarded to the created paths (*Use* state). When the required content is completely downloaded (i.e., all Content Objects received), the corresponding GFL entry is removed (*Expired*).

3.2.3 Path Discovery

The OMP-IF strategy triggers path discovery when an Interest is required to be forwarded, but no matching entry with the *Content Name* is found in the GFL. The Interests that are used to discover paths are referred to as *Probes* and these Interests are broadcast to all the faces in a matching FIB entry (as in the *Standard Strategy* explained in [JST$^+$09]). The broadcast results in the same Interest being replicated

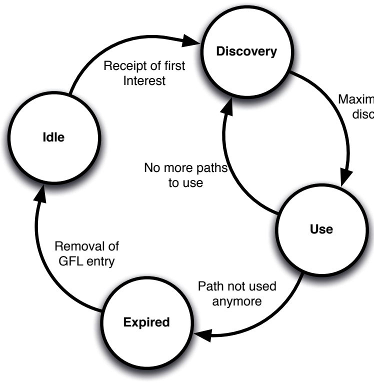


Figure 3.5: States and Transitions of GFL Entries

and therefore, they will carry the same *nonce* value. The intermediate nodes that detect the duplicate *nonces* will only consider the first Interest and discard the rest. As a response to the *Probes*, each node that receives the Interests replies with the Content Objects (provided that the node has the content). There are three possible cases when processing Content Objects.

- A CCN router uses the first arrival Face of the Content Object as the only *good face* in the GFL entry.
- A CCN client places all the Faces through which the copies of Content Objects arrived as *good faces*.
- A CCN router or CCN server (i.e., content producer) replies to each and every Interest received, irrespective of whether the *nonce* is duplicated or not.

This process guarantees the creation of node-disjoint paths from the client to the content locations. Figure 3.6 and Figure 3.7 show how the *Probes* propagate in a CCN network and how the node disjoint multi-path are found when copies of Content Objects are received.

In Figure 3.6, an application executing in node *M* (a client) wishes to download a content (i.e., a file). The application generates the first Interest which is received at the *CCN Layer*. Since there is neither a corresponding GFL entry nor a Pending Interest Table (PIT) entry available, a path discovery is commenced. *M* initiates the path discovery by replicating the Interest into the available faces (based on

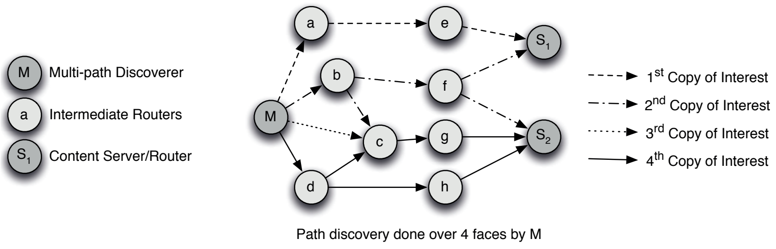


Figure 3.6: Node Disjoint Path Discovery - Interest Flow

FIB entry) which result in intermediate nodes (i.e., CCN routers), *a*, *b*, *c* and *d* receiving these Interests. These routers forward the Interests further based on their *FIB* entries. The router *c* only forwards one of the received Interests due to the duplicate *nonce*. By the time the content locations (i.e., S_1, and S_2) receive these Interests, all the intermediate routers have dropped the "bread crumbs" by means of *PIT* entries.

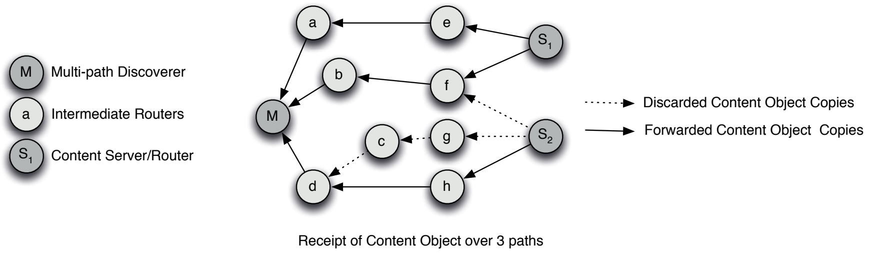

M Multi-path Discoverer
a Intermediate Routers
S1 Content Server/Router
Discarded Content Object Copies
Forwarded Content Object Copies
Receipt of Content Object over 3 paths

Figure 3.7: Node Disjoint Path Discovery - Content Object Flow

As seen in Figure 3.7, S_1 and S_2 replies with a copy of Content Objects for each of the Interests received. These Content Object copies follow the "bread crumbs" and are received at *M*. Though *f* and *d* received multiple copies of Content Objects, they forward only one copy of the Content Object. When the path discovery is complete, *M* would have three paths, *M*-*a*-*e*-S_1, *M*-*b*-*f*-S_1 and *M*-*d*-*h*-S_2 to download the content.

A precondition of OMP-IF is that at most, only one copy of an Interest is forwarded which have duplicate *nonce* values. This precondition is tenable in most cases because as explained in [JST+09], a CCN node only forwards the first copy of an Interest received while the others are simply used to update the PIT and discarded. However, there is an exceptional condition under which an intermediate node (CCN router) may forward multiple copies of Interests with the same *nonce* belonging to the same path discovery process. This exception is explained in the following text.

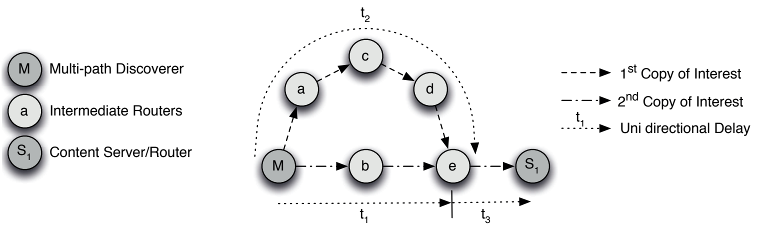


Figure 3.8: Path Delay Asymmetry may Result in Disjoint Path Discovery Failures

As shown in Figure 3.8, the client M sends copies of path discovery Interests over two paths (M-a-c-d-e-S_1 and M-b-e-S_1). When S_1 replies to the Interest over the latter path, the following condition is a possibility due to M-a-c-d-e-S_1 path taking a longer time to deliver an Interest.

$$t_2 > t_1 + (2 \times t_3)$$

where t_1 is the unidirectional delay of path M-b-e, t_2 is the unidirectional delay of path M-a-c-d-e and t_3 is the uni directional delay of path e-S_1.

When this situation arrises, the PIT entry at node e becomes consumed by the first Content Object received and when the second Interest arrives at e, it will simply forward it without knowing that this Interest is part of a currently active path discovery. Therefore, path delay asymmetry results in node e forwarding both Interests.

The consequence of this exception is the creation of an additional path which is inadvertently considered as being a node disjoint path. But this exception is ignored in this thesis as the path usage mechanism (described in Section 3.2.4) minimizes the usage of this path due to the higher latency.

Further, if node e caches Content Objects, then the reply for the second Interest is generated by node e. In such a case too, the higher latency results in the lower usage of that path.

3.2.4 Path Utilization

The path discovery process results in building the GFL entry for the content being downloaded. Once the GFL entry is created and at least one *good face* is available, the OMP-IF strategy will forward the subsequent Interests over the *good faces*. As indicated in the *Design Goals* (Section 3.2.1), the OMP-IF strategy uses splitting of the stream of Interests of a content to the *good faces*. Splitting refers to the distribution of Interests into the multiple *good faces* so that a set of Interests go over a certain face while the others go over other faces. This thesis identifies two possible mechanisms that could be used to identify how the Interests are distributed. They are, Weighted Round Robin (WRR) and Leftover Capacity Utilization (LCU) which are explained in the following sections.

3.2.4.1 Weighted Round Robin (WRR)

Round Robin (RR) is a commonly used scheduling mechanism for data packets or jobs in computer networks. In the simplest form of RR, each process is given an equal opportunity to use a common resource in a cyclical order assigning equal priority (i.e., equal weight) to each process. The WRR goes beyond the RR to assign differing weights to the processes when using the common resource. OMP-IF uses the WRR mechanism when distributing Interests to multiple *good faces* using the Round Trip Time (RTT) as the basis of weight determination (Figure 3.9). The WRR mechanism has two activities in its operation.

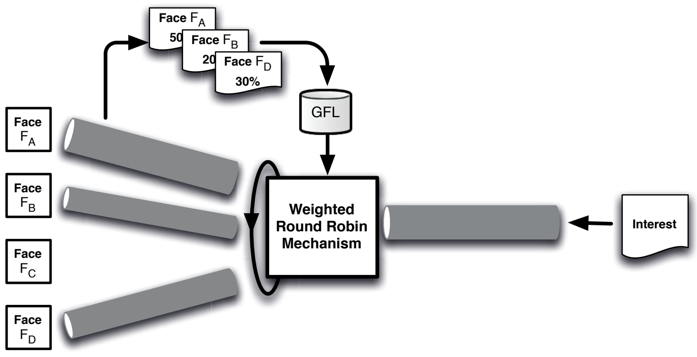


Figure 3.9: WRR - Weights are used to distribute Interests

The **RTT Determination** activity is where the RTT of each path is computed. The RTT is computed by obtaining the time difference for each Interest-Content

Object pair sent and received over a *good face*. The **Weight Determination** activity uses the RTT values computed for each *good face* to identify the weight assigned to each of the *good faces*. The weights are determined using the inverse of the RTT.

Assuming that a client has *G good faces* discovered through path discovery and the current RTT of an Interest-Content Object pair processed by *good face i* (*i = 0, 1, ...G*) is T_i. Then the weight of Face *i* is $\frac{1}{T_i}$.

The weights are computed every time a Content Object is received and when the next Interest is required to be sent out, the OMP-IF strategy randomly selects a face from those *G good faces* according to their weights. Therefore, the probability of *good face i* being selected to deliver the next Interest is,

$$\frac{1}{T_i \sum_{n=1}^{G} \left(T_n^{-1}\right)}$$

In this way, the *good faces* with lower RTT values are assigned higher weights as they are capable of retrieving the content faster. The initial weights for each path is determined from the *Probe* used for path discovery (Section 3.2.3).

3.2.4.2 Leftover Capacity Utilization (LCU)

The WRR mechanism is considered as a performance dependent scheduling mechanism as it adjusts the distribution of Interests based on the RTT. Another performance dependent mechanism is to use more of the *good faces* that are bringing back Content Objects for previously sent Interests, compared to the other *good faces* which are waiting for Content Object arrivals. This mechanism is identified in this thesis as the Leftover Capacity Utilization (LCU) scheduling mechanism (Figure 3.10).

The notion behind this logic is, "if a face is able to bring back Content Objects, then the path associated with that face is capable of bringing more". The basic operation of LCU is to send the next Interest over the path through which the last Content Object was received. The following sequence describes the procedure followed by LCU.

1. Assuming that (based on Figure 3.10),
   - a Content Object arrived at t_1 over face F_B of a particular node,
   - an Interest was received by the same node at t_2 over another face (not F_B) and,
   - no Content Object is received between t_1 and t_2 then,

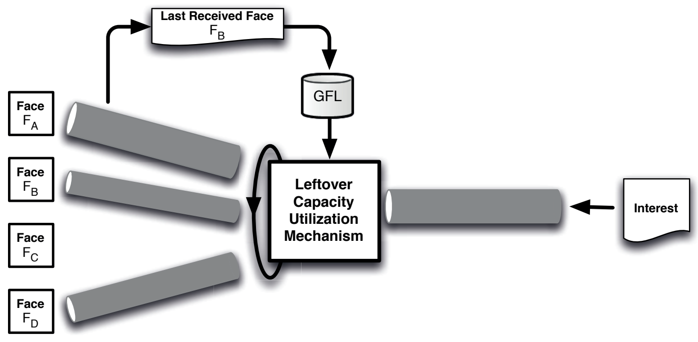


Figure 3.10: LCU - Face through which the last Content Object was received is used to send the next Interest

2. Send the Interest received at t_2 over face F_B.

In this mechanism, all the *good faces* are used to send Interests out, initially. Content Object receipts for these Interests determine the frequency of the use of each *good face* for subsequent Interest.

The operation of these two distribution mechanisms (WRR and LCU) is possible only when the GFL has more than one *good face*. Therefore, these mechanisms operate only at CCN nodes that have multiple faces and is deployed with OMP-IF strategy to perform multi-path content retrievals. In the case of a client focused multi-path deployment, the CCN clients use these mechanisms. In a network based multi-path deployment, the CCN router that performs the multi-path discovery and use employs these mechanisms to distribute the Interest flow.

3.2.5 Path Maintenance

The entries in the PIT point to currently pending Interests for which no Content Objects have been received. When a PIT entry expires, the path over which the corresponding Interest sent is also considered as being invalid. Therefore, the corresponding *good face* in the GFL entry is also removed.

When a *good face* is removed, the OMP-IF strategy will continue with the other *good faces* to forward the subsequent Interests. This is done to avoid the disruption that a content download and the network has to experience when a path re-discovery is attempted. But, by removing the *good face*, if no more *good faces* are left, then the path re-discovery (Section 3.2.3) is started with the first Interest received for forwarding.

3.3 Influences on Caching

Caches in a CCN based network are populated based on the requests for contents made by the CCN clients that request these contents. Therefore, it can be said that, what is in a cache is *as good as what was requested before*. Figure 3.11-(a) shows how a cache would be populated with all the *Chunks* of the two most popular contents. The aspect of what is contained in a cache is influenced by the use of multi-path content retrievals. There are two effects due to these influences. The first of these is called *Cache Fragmentation* while the other is called *Cache Pollution*. *Cache Fragmentation* have a direct relation to multi-path while *Cache Pollution* are a general issue associated with any caching system that also affects the multi-path performance in CCN.

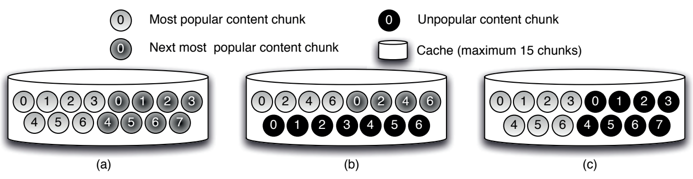

Most popular content chunk
Next most popular content chunk
Unpopular content chunk
Cache (maximum 15 chunks)
(a)
(b)
(c)

Figure 3.11: Effects on Caching - (a) Normal Cache, (b) Fragmented Cache, (c) Polluted Cache

3.3.1 Cache Fragmentation

The OMP-IF strategy discovers multiple paths and when these paths are used, the *Splitting* mechanism is used to distribute the requests to multiple paths. With the *Splitting* mechanism, irrespective of whether the WRR or LCU is used, only a portion of the Interests are sent along a particular path. Since CCN has the concept of "bread crumbs", the Content Objects corresponding to the Interests that travel along a path will also return along the same path. Therefore, the *Splitting* mechanism results in caches along a path being able to only cache a portion of a content and not the whole content, thereby building fragmented caches.

Figure 3.11-(b) shows a cache that is fragmented due to the use of the *Splitting* mechanism. Since the Content Objects for the two most popular contents were requested alternately by a CCN client, the cache has only cached the even numbered *Chunks*. If another CCN client requests the odd numbered *Chunks*, the cache is unable to serve them and therefore must forward the Interests further.

3.3.2 Cache Pollution

Caches usually are populated with most popular contents. What contents are considered as being popular is dependent on the frequency of the requests, the retention policy employed in the cache and the cache size. Assuming that a popularity of content is modeled as a Zipf distribution having 2000 content classes with an α of a 2.0 and a Least Recently Used (LRU) based cache with a storage size of 5000 *Chunks*, mathematically, up to 90% the content *Chunks* can be stored in the cache (Section 6.1.1).

Once a cache is filled based on the popular content, there is a likeliness that requests may arrive for a content that is not considered as being popular. Since an LRU based cache may eject part of the popular content to accommodate the new content, the cache becomes polluted with a content that is unlikely to be requested frequently by others. This is termed as *Cache Pollutions* [RR11]. Figure 3.11-(c) shows a polluted cache where all the *Chunks* of a popular content have been ejected to make way for the unpopular content.

3.3.3 Persistence of Influences on Caching

A cache usually serves a number of CCN clients and the content request pattern of each of these CCN clients may vary according to the popularity of the content. Therefore, even though a single CCN client may fragment a cache due to requesting a part of a content, the request patterns of other CCN clients may result in the cache containing all contiguous *Chunks* of popular content.

In a similar manner, when a popular content is ejected due to the requests of a particular CCN client, the requests of other CCN clients result in this popular content entering the cache once again.

Hence, it could be said that the effect of *Cache Fragmentation* and *Cache Pollution* occurs only in the short term. Due to the actions of other CCN clients (i.e., Zipf based content requests), the long term representation of a cache ultimately will represent the Zipf popularity model and thereby, will hold the most popular contents. The results presented in Section 5.5.4.1 show and discuss the effects of cache fragmentation and cache pollution for contents of popular and unpopular classes when the flow of Interests are split into the multiple paths while using the OMP-IF strategy.

3.4 Related Work on CCN Multi-path Forwarding Strategies

Developing and evaluating multi-path forwarding strategies is a novel area in CCN due to the concepts and mechanisms of CCN being quite recent. The seminal publication on CCN ([JST$^+$09]) was released in 2009. Research related to multi-path extensions for CCN is limited. There are two main publications that addresses the idea of content retrievals with multiple paths where the paths are discovered beforehand. The rest of this section compares and contrasts the work presented in [RR11], [YAM$^+$13] and [YAW$^+$12] with the mechanisms adopted in OMP-IF.

3.4.1 Caching Performance in Multi-path CCN

The authors of [RR11] evaluates the impact on caching when multi-path is used in CCN, through simulations. They consider aspects such as sizing of the content catalogue and caches, content popularity, cache replacement policies and single vs. multi-path forwarding strategies. Based on the overall performance, this work concludes that the impact of the network topology is limited in CCN and that multi-path routing may play against CCN efficiency due to the issues discussed in Sections 3.3.1 and 3.3.2. It further concludes that there is not much impact from the replacement policies employed (e.g. LRU) and the settings of different catalogue sizes and popularity have minor impact.

When considering the evaluations specific to multi-path, this work uses a network with a standard binary tree topology that includes 15 nodes and 8 leaf nodes. The nodes build only two paths (i.e., two faces) and the forwarding strategy employs an Interest replication policy where all Interests are sent over both faces. The comparison is between the *best-face* strategy and multi-path with simultaneous use of paths. The results show that due to the same Interests traveling along multiple paths in parallel, the likeliness of discovering Content Objects increases. However, at the same time this forwarding strategy results is polluted caches. This is the main reason for the authors to conclude that multi-path degrades the performance of CCN. Since the performance evaluation of a cache is meaningful only when the cache represents the popularity model (i.e., steady state), it is not clear whether results in [RR11] were obtained under steady state conditions. Further, the results do not discuss the effect on cache pollution in terms of the popularity of classes as discussed in this thesis (refer Section 5.5.4.1).

As explained in Section 3.3.3 and as seen from the results in Chapter 5, *Cache Pollution* is a short term effect. On the long term, caches tend to possess the Zipf characteristics. Further, the results in Chapter 5 also shows that replicating Interests have a negative effect on the overall performance of CCN compared to distributing the Interests over the multiple paths. Due to replications, the network is flooded with Interests and this drastically increases the congestion and thereby reduces the throughput of clients.

With the OMP-IF strategy, replication of Interests are only done during the path discovery as explained in Section 3.2.3 and once the *good faces* are discovered, the subsequent Interests are distributed to the *good faces* (i.e., with the *Splitting* mechanism) based on the path usage mechanisms described in Section 3.2.4.

3.4.2 Stateful Forwarding Plane in NDN

The authors of [YAM$^+$13] and [YAW$^+$12] working for the Named Data Networking (NDN) Project [ZAB$^+$14] propose a forwarding plane which lets NDN routers keep track of data delivery performance, control network load in the NDN architecture. NDN is a derivative of CCN mainly focussing on extending the basic functionality (such as forwarding, naming conventions, etc.) proposed in CCN.

The work extends CCN to have an Interest Negative Acknowledgment (NACK) which notifies the downstream node (a node towards the client) of network problems quickly, which can then take proper actions based on the error code, and delete the unsatisfiable Interest from the PIT. For example, when an NDN node can neither satisfy nor forward an Interest (e.g., there is no Face available for the requested name), it sends an Interest NACK back to the downstream node. A NACK carries the same content name as in the respective Interest, plus an error code explaining why the NACK is generated (e.g., Congestion, No Path, etc.). If the downstream node has exhausted all its own forwarding options, it will send the NACK further, i.e., downstream. In the absence of packet losses, every pending Interest is consumed by either a returned Content Object or a NACK.

The information (interface is active, RTT estimation, congestion of a link, etc.) carried in the Interest NACK is used to populate the FIB. The work also point to some open discussions in the area of how to react when an Interest NACK is received in case of discovering a new path and the use of multiple paths. The forwarding plane is proposed as a case study and the main objective is to show how an NDN node is able to react dynamically to adapt to network conditions.

The forwarding plane introduced in this work can inter-work with the OMP-IF strategy that is proposed in this thesis. OMP-IF strategy can utilize the information available in the forwarding plane to take decisions dynamically when discovering a new path and using the Interest flow splitting mechanism to distribute the Interests among the multiple paths.

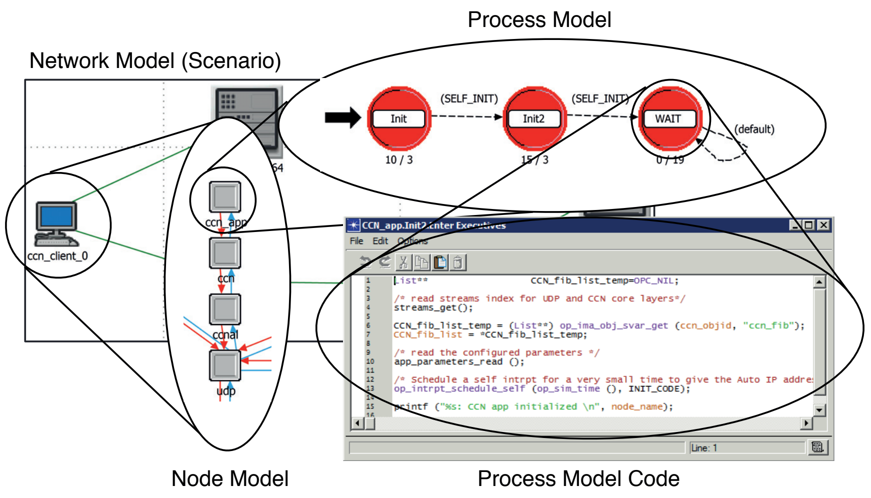
4 Simulation Model

Chapter 3 identified the On-demand Multi-path - Interest Forwarding (OMP-IF) strategy for Content Centric Networking (CCN) and the associated mechanisms. The purpose of this chapter is to detail the simulation model developed to evaluate the performance of OMP-IF. The simulation model was developed using the Optimized Network Engineering Tools (OPNET) environment [GXW07]. The model consists of two different CCN node implementations that contain a number of features, based on the description given in [JST$^+$09]. Each of these nodes have the CCN protocol stack implemented in them for communications and the CCN functionality is overlaid on Internet Protocol (TCP/IP). The chapter starts with a brief description of the OPNET environment. Then, the description moves on to the details of the CCN simulation model that includes detailed descriptions of the node model and the operations of the protocol layers.

4.1 OPNET Environment

OPNET is a well known network simulation environment used in research and teaching [GXW07]. It has two main parts to it; the *modeler* and the *simulation run-time*. The modeler allows the user to build network scenarios by combining existing or self built network elements (such as routers, applications servers, user terminals and links) and the simulation run-time allows the user to execute the created scenarios to gather performance statistics for analysis.

OPNET is a graphical tool where most activities, such as the creation of scenarios, are done using Graphical User Interface (GUI) based editors. These editors allow users to drag-and-drop components to create the network models (scenarios), node models and process models. OPNET provides implementations for a number of different networking elements and their protocol stacks consisting of generic models (e.g., 100BaseT links, Mobile IPv6 Clients) and of specific networking solution provider models (e.g., Cisco C4700 Router, IBM AIX Server). But, OPNET has not provided implementations for each and every network element. CCN is one such example. Therefore, new node types and protocols have to



```
List**                          CCN_fib_list_temp=OPC_NIL;

/* read streams index for UDP and CCN core layers*/
streams_get();

CCN_fib_list_temp = (List**) op_ima_obj_svar_get (ccn_objid, "ccn_fib");
CCN_fib_list = *CCN_fib_list_temp;

/* read the configured parameters */
app_parameters_read ();

/* Schedule a self intrpt for a very small time to give the Auto IP addre
op_intrpt_schedule_self (op_sim_time (), INIT_CODE);

printf ("%s: CCN app initialized \n", node_name);
```

Figure 4.1: OPNET Editors: Network Model Editor, Node Model Editor, Process Model Editor and the Process Model Code Editor

be implemented to handle CCN functionality by writing the necessary code. The general workflow of using OPNET is as follows.

1. Create a network model (scenario) by combining different networking elements
2. Select the statistics that need to be collected
3. Execute the simulation
4. Analyze the collected statistics

The **Network Model** editor allows a user to create networks by combining network elements and links. Each network element is made of a number of different protocol layers. The combining and linking of these protocol layers is done using the **Node Model** editor. Each of the protocol layers is a **Process Model** which is a state machine represented by a Finite State Machine (FSM). The states and the transitions of the FSM consist of code that performs the operations of the protocol layer. The code, based on the C language is written using a **Process Model Code** editor (Figure 4.1). The numbers below each state identifies the number of lines of code in each state. There are two types of states in OPNET. The *Unforced* state (red) is a state where the control stays in the state when the state is reached while the control in a *Forced* state (green) retunes to an Unforced state after the code in the Forced state is executed.

4.2 Protocol Stack

The OPNET based CCN model, implemented as part of this thesis, has three types of node models, viz., CCN client, CCN router and CCN server. All of these node models are deployed with the CCN protocol stack but serving different purposes. A CCN client initiates content downloads and has mobility support. A CCN server hosts the content requested by CCN clients, similar to a web server. A CCN router performs the task of forwarding Interests and Content Objects, and caching Content Objects.

To perform the functionality described in [JST$^+$09], the protocol stack deployed in each of these node models implement the following layers.

- **Application Layer**: Implements the instantiation and execution of applications such as file downloading or streaming
- **CCN Layer**: Implements the CCN functionality that mainly consists of forwarding and caching
- **Adaptation Layer**: Implements the transformations required between the CCN layer and the underlying TCP/IP layer
- **Transport Layer**: Implements the User Datagram Protocol (UDP) based transport layer over which CCN communications occur
- **Network Layer**: Implements the Internet Protocol (IP) layer
- **MAC Layer**: Implements the link technology used for communications between the different nodes based on a Media Access Control (MAC) protocol.

These layers and the functionality are described in detail in the subsequent sections. The six layered protocol stack is used in the different node models as shown in Figure 4.2.

A CCN client node model in the simulator implements all the layers of the CCN protocol stack. These node models execute applications and do not act as routers (i.e., no forwarding of CCN Interests and Content Objects received from other CCN nodes). Further, they do not provide support for caching of content at the CCN layer.

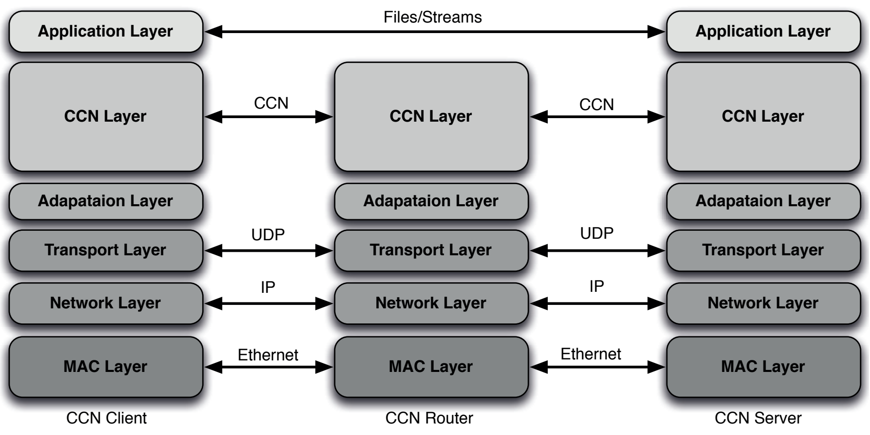


Figure 4.2: CCN Protocol Stack in Nodes

The CCN router node models on the other hand have the capability to perform caching and forward Interests and Content Objects based on the forwarding strategy employed. Figure 4.3 shows a partial OPNET scenario with a CCN client possessing multiple Faces connected to CCN Routers.

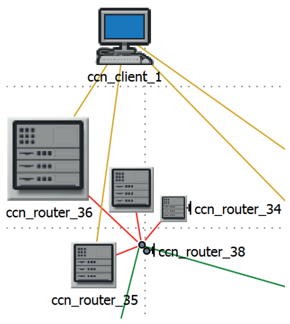


Figure 4.3: A cross section of an OPNET scenario showing an inter-connected CCN Client (*ccn_client_1*) and CCN Routers

Each of the above protocol layers are implemented using OPNET process models. Figure 4.4 shows the node model of a CCN client with 2 network attachments together with the process models handling the individual protocol layers.

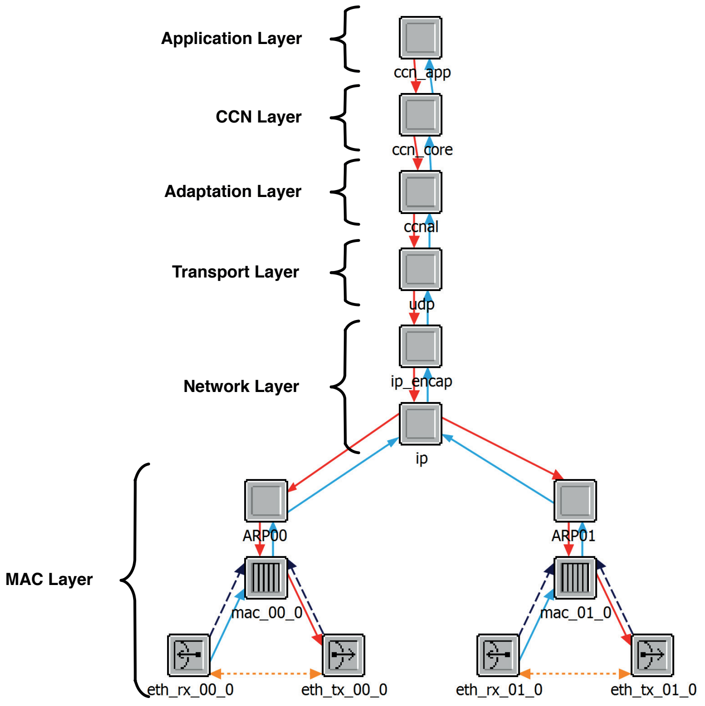


Figure 4.4: Node Model of CCN in OPNET identified by Protocol Layer

4.3 Application Layer

The application layer handles the user applications that request, consume and serve content. There are two categories of process models that operate at the application layer. They are,

- Application Management Process Model
- Application Process Models

The Application Management Process Model (*ccn_app*) is a process model that manages the instantiation of application type process models. A *ccn_app* retrieves the user parameters and passes them to the instantiated application type process model. Application type process models are the individual process models that perform the activities related to content retrieval (CCN client) or serving content

(CCN server). When configuring the network model (scenario), the attributes of each these node models are configured to describe the deployment of the application process models.

There are two types of application process models that can be deployed, one in CCN client type node models while the other is in CCN server type node models. They are called Client Application Process Model (*client_app*) and Server Application Process Model (*server_app*). The details are given below.

4.3.1 Client Application Process Model (*client_app*)

The Client Application Process Model (*client_app*) is the process model that models the operation of a user application to request for content, which is deployed in a CCN client. The *client_app* generates Interests based on the configuration and accepts Content Objects. Figure 4.5 shows the FSM of *client_app*. The FSM consists of the following states and transitions.

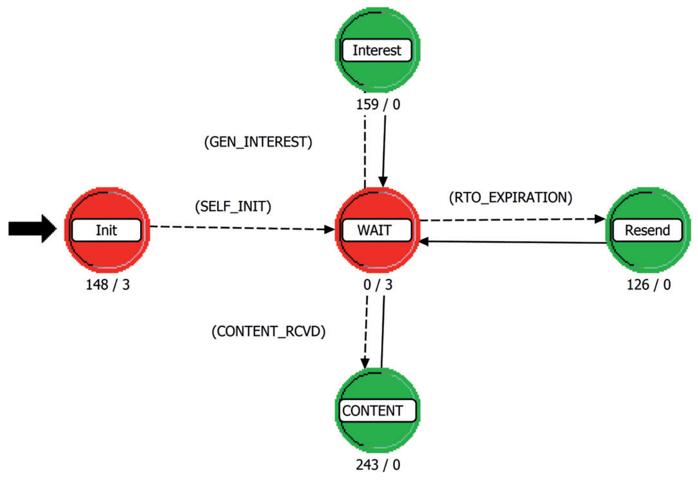


Figure 4.5: FSM of the Client Process Model

- **Init**: The Init state is the entry state in the FSM (black arrow). This state obtains the parameters passed, performs the initializations to the Zipf Cumulative Distribution Function (CDF) to generate content based on popularity, initializes the statistics to be collected and sets up a timer to download the first content. The Init state also initializes the Flow and Congestion Control (F&CC) state for the first content download. Once all the these actions are done, a transition will occur to the Wait state.

- **Wait**: The Wait state is the state where the client application waits for any event to occur to perform the transition in Figure 4.5). The possible events are, timer expiration (*GEN_INTEREST*) to generate the the first Interest to send, receipt of a Content Object (*CONTENT_RCVD*) or an expiration (*RTO_EXPIRATION*) of a Pending Interest Table (PIT) entry. Once any of these events occur, control is passed to either Interest, Content or Resend states.

- **Interest**: The move to the Interest state occurs when the timer for the start of the first content download expires. This expiration results in the generation of the first Interest. Additionally, the Content state forces control to pass to this state when Interests need to be sent due to the receipts of Content Objects. The receipt of a Content Object results in the freeing up of the F&CC window which in turn makes it possible to send more Interests.

- **Content**: The move to the Content state occurs when a Content Object is received for processing. This state updates F&CC parameters and control is forced to go to the Interest state to send the next Interests.

- **Resend**: The Resend state gets invoked when Interests need to be expired due to not receiving the corresponding Content Objects.

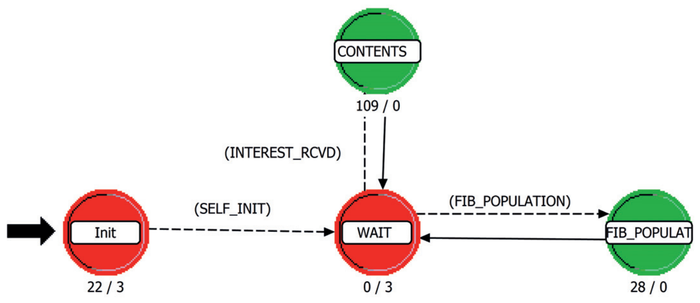


Figure 4.6: FSM of the Server Process Model

The Interest, Content and Resend states collect statistics related to the application layer.

4.3.2 Server Application Process Model (*server_app*)

The Server Application Process Model (*server_app*) is the second of the application type process models which is deployed in a CCN server. The purpose of *server_app* is to respond to content download requests by replying with Content Objects. Figure 4.6 shows the FSM of *server_app*. The FSM consists of the following states and transitions.

- **Init**: The Init state is the entry state when the *server_app* is instantiated. The purpose of the state is to initialize the statistic collection variables and setup a time to expire for the Forwarding Information Base (FIB) population to occur.

- **Wait**: The Wait state is the state where the process model resides expecting an event to occur. The possible events are, expiration of the FIB population timer (*FIB_POPULATE*) or the receipt of an Interest (*INTEREST_RCVD*).

- **Contents**: The Contents state is the state that is invoked in response to an Interest, to reply with a Content Object. All received Interests are replied with Content Objects as the CCN server is considered to host all the Chunks of a content.

- **FIB Populate**: The FIB Populate state is the state that is invoked at the beginning of any simulation to propagate prefix served by the *server_app* throughout the CCN network. The prefix propagation is done using a Prefix Advertisement (PA) type message. This message is neither an Interest nor a Content Object. The PA is used in the same manner as a Link-State Advertisement (LSA) is used in Section 2.7.6. Using this information, all the CCN routers populate their FIBs with the prefix and the Faces through which the prefix is reachable.

CCN is a receiver driven networking environment [JST$^+$09]. Therefore, the task of the *server_app* is limited to replying to the requests for contents (i.e., no connections established). In contrast, *client_app* controls all communications in CCN. To enable a realistic communication model in the CCN networks simulated, the *client_app* is included with a traffic generation model based on the Zipf popularity model [Wik35] and an adapted F&CC model based on the model used in the Transmission Control Protocol (TCP) protocol in TCP/IP as described in [UCG13] by the author of this thesis.

4.3.3 Traffic Generation Model

The *client_app* uses a Zipf based popularity model when generating requests for contents. [GDS$^+$03] and [CKR$^+$07] has shown that traffic in the Internet have characteristics similar to the Zipf model. Further, the work done in [MCG11], [CGLD11] and [RR11] has adopted the Zipf based traffic generation model in the context of evaluating the performance of CCN based networks.

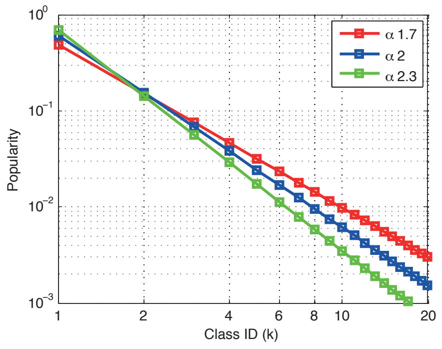


Figure 4.7: Zipf popularities for different α values

In a Zipf based popularity model, contents are grouped into numbered classes and a given class is said to be more popular than the immediately following class. This relationship is mathematically stated in the following manner.

- Assuming the existence of a content catalog of M content items equally partitioned into K classes based on popularity, content items of class k is requested with a probability of q_k as represented by the Equation 4.1,

$$q_k = \frac{c}{k^\alpha} \tag{4.1}$$

  where k is the class number, α characterizes the form of the popularity curve i.e., the form of the tail of the curve and c is the normalization factor computed as $c = \left(\sum_{k=1}^{K} \frac{1}{k^\alpha}\right)^{-1}$. Figure 4.7 shows the Zipf popularity curves of the first 20 classes with differing α values, drawn in log scale. The curves show Zipf distributions with higher α values have a small number of extremely popular content classes while with lower α values, the popularity is spread over more content classes.

- Due to a total of M contents and K classes, each class has $\frac{M}{K}$ contents per class. The contents within a class are uniformly popular and the probability of requesting a single content within a class, i.e., p_b is represented by Equation 4.2,

$$p_b = \frac{K}{M} \tag{4.2}$$

  where b is the number of the content within the class k.

- Based on the above explanations, the probability of requesting a single content with a given class (i.e., k) is represented by Equation 4.3,

$$p_{k,b} = q_k p_b \tag{4.3}$$

- The sizes of content in terms of Chunks is considered to be geometrically distributed (explained in Section 6.1.2) with a probability of p_l and the mean Chunk size of a content, ω is represented by Equation 4.4,

$$\omega = \frac{1}{p_l} - 1 \tag{4.4}$$

The *client_app* creates a CDF of the Zipf probabilities (i.e., q_k) to use when requesting the next content to download. Every content name is encoded with the Zipf information and which content to download is identified using two uniformly distributed random numbers. The first of these random numbers is used to determine the Zipf class of the content to be downloaded. The class is picked from the Zipf CDF created. The second of these numbers is used to determine the content number within the class. Based on the above description, an example content name of a requested content by the *client_app* is as follows.

/unibremen/video-*x*-*y*.avi

where $x \in \{1,2,3,\ldots,K\}$ and $y \in \{1,2,3,\ldots,\frac{M}{K}\}$. Due to the nature of the Zipf popularity model on which x is based, content of lower numbered classes (i.e., popular classes) are requested more than the content of higher numbered classes. *client_app* determines the contents to download in a sequential manner, i.e., immediately after the download of all the chunks of a content is completed, it creates another content name using x and y to download. A content to be considered as being completely downloaded, all Interests associated with that content must have been sent out by the *client_app* and all the corresponding Content Objects must have been received by the *client_app*. When PIT expirations occur, the corresponding Interests are resent by the *client_app*.

The *server_app* replies to any Interest that it receives with a Content Object, provided that it has the **/unibremen** prefix in the content name.

4.3.4 Flow and Congestion Control (F&CC)

The applications in CCN are receiver driven. As detailed in Section 2.7.1, this means that the user must request for a content using Interests. The Content Objects arrive only in response to Interests. In contrast, TCP based networks has the sender of a communication session mainly controlling the flow of data. TCP is the most widely used F&CC protocol in the world due to the Internet. In this thesis, a receiver driven TCP-like Flow and Congestion Control (F&CC) mechanism is implemented in the *client_app*. The main purpose of F&CC in CCN is to control the flow of the requested Content Objects. By controlling the flow, CCN is able to avoid the negative effects of congestion in the network and also the effects of differing distances to the caches.

An F&CC mechanism that extends the operations described in the *NewReno* TCP flavor ([Flo04], [Ste97], [APS99], [PACS11]) is used in the *client_app*. The following list provides the aspects considered and the details of the extensions, contrasting with TCP.

- **Control/Communication Orientation**: In TCP, the sender of data in a communication session is in control of the session rather than the receiver. This means that the sender continuously sends data to the receiver based on the sender's view on how the network performs. On the other hand, no data will traverse the network unless the data has been requested for in CCN and therefore, in CCN, the receiver is in control. This means that the receiver is in charge of the data flow by controlling how Interests are sent.

- **Data Acknowledgment**: In TCP, data are acknowledged by a message that travels in the opposite direction, to the sender. This TCP Acknowledgement Message (ACK) mechanism is one of the fundamental pillars of TCP, used by the sender to make a number of decisions on how the rest of the data must be transmitted. But in CCN, there is no concept of ACKs. Therefore, the receipt of a Content Object itself is considered as the acknowledgement in CCN.

- **Congestion Window Management**: One of the key aspects of TCP is the use of a congestion window to control the flow of data between the sender and the receiver. Due to the sender orientation of TCP, a sender maintained window controls the communication session. On the other hand, since CCN is receiver oriented and the Interest sending is considered as the means by which the flow of data is controlled, the congestion windows is maintained at the receiver.

- **Congestion (Packet Loss) Detection**: TCP uses the Re-transmission Timeout (RTO) and the receipt of 3 duplicate ACKs as the basis for considering congestion in the network [APS99]. In the case of CCN, the RTO can be built in the same way as in TCP by considering the Round Trip Time (RTT) associated with the Interest-Content Object cycle. But, CCN does not have the concept of 3 duplicate ACKs and further, unlike TCP, CCN has the additional problem of determining whether any out-of-order Content Object receipts are due to congestion (or packet loss) in the network or due to Content Objects arriving from multiple sources. Therefore, the implementation uses an algorithm called *Pre-Recovery* that addresses the issue associated with multi-source arrivals [UCG13].
- **Recovery Algorithm**: TCP uses the Fast Recovery and Fast Retransmit algorithms to resend data which are considered to be lost. Since CCN does not have the concept of duplicate ACKs, the simulator uses a CCN based *Fast Recovery* algorithm and *Selective Fast Retransmit* algorithm to handle the resending of Interest packets.

A brief explanation of the *Pre-Recovery*, *Fast Recovery* and *Selective Fast Retransmit* algorithms detailed in [UCG13] is given in Section 4.3.4.2.

4.3.4.1 F&CC Parameters

The F&CC in CCN is based on the combined operation of 5 algorithms: *Slow Start*, *Congestion Avoidance*, *Pre-Recovery*, *Fast Recovery* and *Selective Fast Retransmit*. The operation of these algorithms require the use of a set of variables. Some of these variables have the same meaning as in TCP [PACS11], but for clarification, they are detailed below in the context of CCN.

- **Chunk/Segment Size** (*seg_size*): The *seg_size* holds the size of the payload field of a Content Object in bytes. This value is a constant maintained by the *server_app* and is made aware at the *client_app* through the first Content Object sent by the *server_app* in response to an Interest sent by the *client_app*.

- **Congestion Window Size** (*cwnd*): The *cwnd* holds the total number of bytes that may be requested by sending Interests. This value maybe higher than all the bytes requested through currently pending Interests (i.e., value in *flight_size*). The *cwnd* is updated during the operations of the F&CC algorithms.
- **Slow Start Threshold** (*ssthresh*): The *ssthresh* holds the limit in bytes of the *cwnd* can reach after which the *Congestion Avoidance* algorithms takes over. When *cwnd* is under the *ssthresh*, the *Slow Start* algorithm is in charge of increasing the size of *cwnd*. The *ssthresh* is updated during the operations of the F&CC algorithms.
- **Re-transmission Timeout** (*rto*): The *rto* is the time in seconds after which a pending Interest is considered to be lost.
- **Round-trip Time** (*rtt*): The *rtt* holds the time difference between the departure time of an Interest generated by an *client_app* and the received time of the corresponding Content Object. This value is used in computing the *rto* in the F&CC algorithms.
- **In-flight Size** (*flight_size*): The *flight_size* holds the number of bytes requested through Interests that have not been received yet. The value in *flight_size* is always less than or equal to *cwnd*.
- **Total Interests to Dispatch** (*dispatch_size*): The *dispatch_size* holds the number of Interests that are to be sent out after any adjustment is done to the *cwnd*.
- **Out-of-order Count** (*ooo_count*): The *ooo_count* holds the number of Interests that are out-of-order during the *Pre-Recovery* phase.

4.3.4.2 F&CC Algorithms

Similar to TCP, CCN maintains a sliding window to control the flow of a communication session. But, unlike TCP, the sliding window in CCN is maintained and controlled at the receiver side (by the *client_app*), and the sender does not have any explicit influence on the control of this window. This sliding window (called the *Congestion Window* as in TCP) controls the flow of Interests originating from the *client_app*. As stated in Section 4.3.4.1, there are 5 algorithms associated with F&CC and operating at the *client_app*, that is tasked with pipelining Interests and taking appropriate action when perceived congestion occurs in the network. These algorithms are similar in how they operate under TCP, but they are described below in the context of CCN.

- **Slow Start**: The *Slow Start* algorithm is active (i.e., *Slow Start Phase*) when the size of *cwnd* is less than or equal to *ssthresh*. A receipt of a Content Object during this phase by *client_app* in response to the corresponding Interest sent previously results in the *cwnd* being incremented by *seg_size*. As a result of the receipt of the Content Object and the increase of the *cwnd*, two Interests are dispatched by the *client_app* as shown by the following equations.

  Receipt of a Content Object results in adjustments to *cwnd* and *flight_size* as shown in Equations 4.5.

  $$\begin{aligned} cwnd &= cwnd + seg\_size \\ flight\_size &= flight\_size - seg\_size \end{aligned} \tag{4.5}$$

  Due to the above adjustments, *client_app* is able to send *dispatch_size* Interests and adjust the *flight_size* as shown in Equations 4.6.

  $$\begin{aligned} dispatch\_size &= floor\left(\frac{cwnd - flight\_size}{seg\_size}\right) \\ flight\_size &= flight\_size + (seg\_size * dispatch\_size) \end{aligned} \tag{4.6}$$

  Due to the above computations, the *cwnd* has an exponential growth during the *Slow Start Phase*.

- **Congestion Avoidance**: The *Congestion Avoidance* algorithm becomes active (i.e., *Congestion Avoidance Phase*) when the size of *cwnd* is larger than *ssthresh*. Content Objects received during this phase results in the fractional increase of *cwnd*. The *cwnd* and the *flight_size* adjustments occurs as shown in Equations 4.7.

  $$\begin{aligned} cwnd &= cwnd + \left(\frac{seg\_size}{cwnd} * seg\_size\right) \\ flight\_size &= flight\_size - seg\_size \end{aligned} \tag{4.7}$$

Due to the fractional increase of *cwnd* and hence, the increase being insufficient for a complete *seg_size*, it may only be possible to send one Interest out after every RTT.

- **Pre-Recovery**: The *Pre-Recovery* algorithm becomes active when 3 out-of-order Content Objects are received (i.e., when Fast Retransmit Threshold (FRT) is reached). It is similar to the receipt of 3 Duplicate ACKs in TCP, but does not proceed to *Fast Recovery* until the dynamically computed Pre_FRT limit for out-of-order Content Objects is reached as described in [UCG13].

  The reason for using the *Pre-Recovery* is due to the unique nature of CCN where out-of-order Content Objects may be due to not only losses or delays but also due to multi-source arrivals of Content Objects due to in-network caching. The *Pre-Recovery* algorithm determines the point at which the *Fast Recovery* and *Selective Fast Retransmit* algorithms are activated. The update to the *cwnd* occurs as performed in the *Congestion Avoidance Phase*.

  If the arrivals of Content Objects return to the state of in-order receipts, *Pre-Recovery* is exited and *Congestion Avoidance* takes over.

- **Fast Recovery and Selective Fast Retransmit**: The *Fast Recovery* and *Selective Fast Retransmit* algorithms are entered into when *Pre-Recovery* determines that a likely loss of an Interest or Content Object has occurred (i.e., out-of-order Content Objects reach FRT + Pre_FRT limit). On entering *Fast Recovery*, the *cwnd* is set as in Equation 4.8.

$$cwnd = ssthresh + (seg\_size * ooo\_count); \tag{4.8}$$

  An out-of-order receipt of a Content Object during *Fast Recovery* results in the increase of *cwnd* by one *seg_size*. *Selective Fast Retransmit* is where the Interests for which no Content Objects were received are identified and resent by *client_app*. Once all the Content Objects are received for the resent Interests and the receipts are in-order, the *cwnd* is set to *ssthresh*.

Every Interest sent has an expiration time based on RTO. The RTO used to set this timer is computed every time a Content Object is received, using the received time with the sending time of the corresponding Interest. If an RTO expiration occurs during the execution of any one of the above algorithms, the *cwnd* is set to *ssthresh*. The computation of the RTO (i.e., *rto*) is done in the same manner as in

TCP, described in [PACS11]. Figure 4.8 shows the congestion window behavior during the operation of the above algorithms.

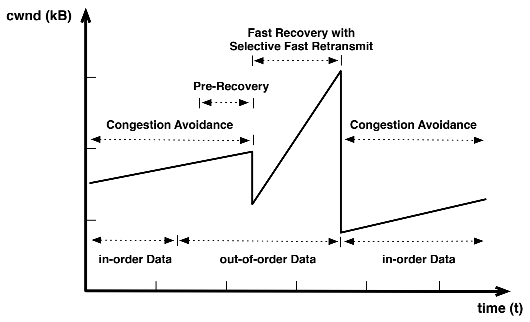


Figure 4.8: Adapted Flow and Congestion Control (F&CC) Algorithms

4.4 CCN Layer

The CCN layer is the layer that handles all core functionality of CCN. It is tasked with processing the arrivals of Interests, Content Objects and PAs from the upper (Application) or lower (Adaptation) layers and once processed, forwarding them to either the lower or upper layers again. The activities include making forwarding decisions, caching content, handling mobility related updates and status maintenance activities.

4.4.1 CCN Process Model (*ccn_core*)

The CCN Process Model (*ccn_core*) implements a subset of the CCN functionality described in [JST$^+$09] and [Mos14]. Figure 4.9 shows the FSM of *ccn_core*. It consists of the following states and transitions.

- **Init** and **Init2**: The Init and Init2 states are responsible for initializing the variables (state variables, statistics, cache storage, etc.), reading configuration parameters and commencing the simulation. The activities done by these 2 states have to be placed in 2 separate states due to CCN Process Model (*ccn_core*) requiring access to information at the IP layer (to obtain Face information). Due to the delays in IP layer initializations during simulations, *ccn_core* is required to access IP layer information with a delay. Therefore, the Init2 state is invoked with a delay of 100 milliseconds.

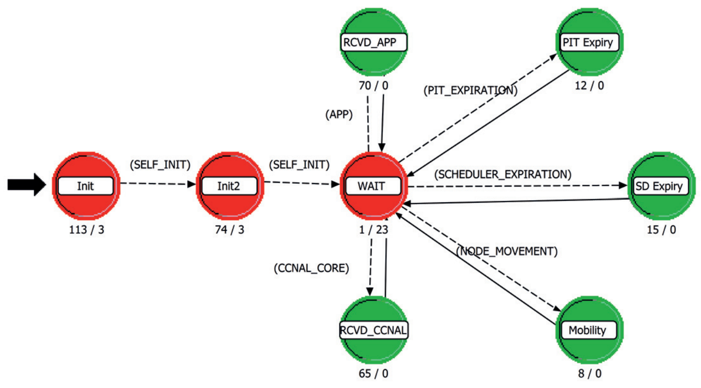


Figure 4.9: FSM of the CCN Layer Process Model

- **Wait**: The Wait state is where the *ccn_core* waits for any event to occur. The possible events are the arrival of an Interest or Content Object from the application layer (*APP*), arrival of an Interest or Content Object from the CCN Adaptation layer (*CCNAL_CORE*), expiration of a PIT entry (*PIT_EXPIRATION*), expiration of a Good Face List (GFL) entry (*SCHEDULER_EXPIRATION*) or the notification of a mobility event (*NODE_MOVEMENT*). When any one of these events occur, control is transitioned to the corresponding Forced states (*RCVD_APP*, etc.) to handle the event and then control returns back to the Wait state.

- **RCVD_APP**: The RCVD_APP state is transitioned to when an Interest, Content Object or Prefix Announcement arrives from the application layer (i.e., either from a *client_app* or a *server_app*) to be processed and routed. Once the processing of these packets are done, they are passed to the CCN Adaptation layer to forward out of the CCN node.

- **RCVD_CCNAL**: The RCVD_CCNAL state is transitioned to when an Interest, Content Object or Prefix Announcement arrives from the CCN Adaptation layer (i.e., from the lower layers) for processing and forwarding. Once the processing is done, these packets are sent either to the applications (i.e., *client_app*s or *server_app*s) or to the Adaptation layer to be forwarded out of the CCN node.

- **PIT Expiry**: The PIT Expiry state is invoked to perform the required processing when PIT entries expire. There are two types of expirations (soft and hard) that have to be taken care of by the functions executed while in this state (see Section 4.4.5).

- **SD Expiry**: The SD Expiry state is invoked to perform the required processing when GFL entries expire.
- **Mobility**: The Mobility state is invoked to handle the processing required when node mobility occurs. The simulator uses a proactive mobility mechanism where the CCN layer is notified of link breaks from the lower layers. When these notifications arrive, the CCN layer updates the different status information (such as the GFL and PIT) to minimize the disruptions to the active content downloads.

The CCN layer, through the *ccn_core* implements the CCN node architecture shown in Figure 2.8 using memory based lists for each of the information stores (FIB, PIT and Content Store (CS)). Each CCN node is capable of having multiple network attachments and each of these attachments is considered as a Face. The *ccn_core* implements a number of mechanisms to handle the operations described in Section 2.7. The following sections describe these mechanisms.

4.4.2 FIB Population

One of the tasks of the *server_app* is the generation of Prefix Advertisements (PAs) to notify a network of the prefixes served by the *server_app*. Though these PAs are created by the *server_app*, they are propagated and processed by the *ccn_core*. Following are the details of processing and propagation.

- **Processing**: When a *ccn_core* receives a PA, the FIB is checked to see if the prefix in the PA exist. If the prefix exists in the FIB, this entry is added with the arrival Face of the PA. If not, a new FIB entry is created with the prefix and the arrival Face. The update of the FIB is done irrespective of whether the node is a client node or a router node.
- **Propagation**: Once the updates to the FIB are done, the PA is forwarded further. The forwarding of a PA is only done by router type nodes. Further, if the PA had been seen before, it is not forwarded further. To forward, the *ccn_core* uses all the Faces, except the arrival Face.

4.4.3 Forwarding Strategies

The simulator implements the *Best-face* and the *OMP-IF* forwarding strategies. Both strategies, in general, have similar operations. Figure 4.10 shows the high level flow charts of processing an Interest (left) and processing a Content Object (right) at a CCN client node or a router node.

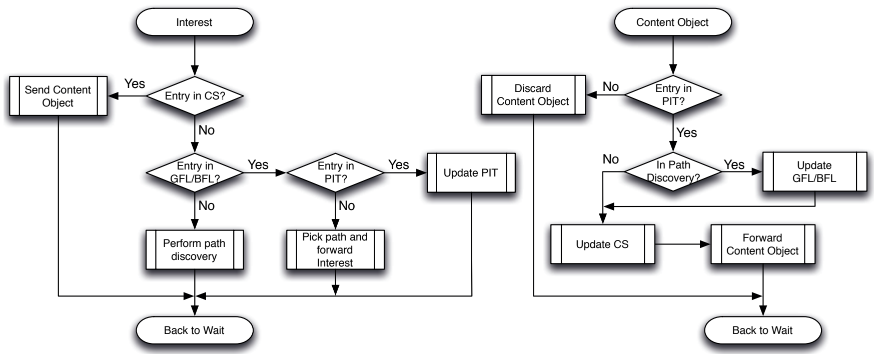


Figure 4.10: Interest Processing (left) and Content Object Processing (right) Flow Charts of best-face and OMP-IF Forwarding Strategies

When an Interest is received, the CS is consulted to check the existence of the Chunk being requested (*Entry in CS?*), if caching is enabled in the node. Section 4.4.4 discusses the details of the caching implementation in the simulator. If the Chunk exists, a Content Object is generated and sent over the Interest arrival Face. If the paths were not discovered before for the content being requested (*Entry in GFL/BFL*), then a path discovery is initiated (*Perform path discovery*). Sections 4.4.3.1 and 4.4.3.2 provides details on path discovery in the simulator. If the Interest was seen before (*Entry in PIT?*), only the Face information is updated in the PIT entry (*Update PIT*). Details on managing the PIT in the simulator is described in Section 4.4.5. If the Interest is seen for the first time, the Interest is forwarded after selecting a path (*Pick path and forward Interest*). Details of the Interest distribution mechanism used in the simulator is described in Section 4.4.6.

When a Content Object is received, the PIT is checked to see if a corresponding entry exists (*Entry in PIT?*). Unsolicited Content Objects (i.e., Content Objects for which no Interest was received before) are always discarded (*Discard Content Object*). If a path discovery is active (*In Path Discovery?*), path information is updated in the GFL or Best Face List (BFL), depending on the forwarding strategy. Sections 4.4.3.1 and 4.4.3.2 provide details on path discovery in the simulator. Once a valid Content Object is received, it is updated in the CS (*Update CS*), if caching is enabled in the node. Section 4.4.4 discusses the details of caching implementation in the simulator. The Content Object is then forwarded in the direction of the requester based on the information in the PIT entry (*Forward Content Object*). The PIT entry is usually removed (i.e., consumed) due to the Content Object but, the entry may continue to be in the PIT if path discovery is active (Sections 4.4.3.1 and 4.4.3.2).

4.4.3.1 OMP-IF Strategy

When a GFL entry is not found, CCN commences path discovery by finding an appropriate FIB entry and replicating the Interest to all the Faces in the FIB entry. Once the Interest is forwarded, the PIT is updated to indicate the expected arrival of the Content Object. The operation for a CCN client node and a CCN router node differs when creating PIT entries. Due to the expectation of multiple copies of a single Content Object over the multiple paths, a client node does not remove the PIT entry on the receipt of the first Content Object. The *ccn_core* on a client continues to maintain the relevant PIT entry until the configured number of paths have been discovered. Even though a client node receives multiple copies, only the first copy of the Content Object is sent to the application. The others are only used to update path information by the *ccn_core* and then they are discarded.

On a router node, the first Content Object receipt results in the removal (i.e., consumption) of the PIT entry. This is similar to the operation of the *Best-face* strategy.

4.4.3.2 Best-face Strategy

The simulator implements the Best-face strategy as explained in [JST$^+$09]. Similar to OMP-IF, the Best-face strategy also replicates the Interests to the multiple Faces during path discovery. Though subsequently, there is the possibility of receiving multiple copies of the same Content Object, the *ccn_core* only considers the path over which the first Content Object was received. The Interests that arrive later for the same content uses this path to forward the Interests (if the chunk is not available in the CS).

4.4.4 Content Store (CS) Management

The CS implemented in the *ccn_core* is configurable to be used in any type of CCN node (i.e., client or router). Usually clients have limited or no caching at all due to the performance expectations of a cache. Caches in CCN are required to perform cache operations at network speeds (i.e., line speeds). Therefore, caches require resources to perform line speed searches and updates that could only be performed by dedicated devices such as powerful routers which can double as caches.

There are 2 policies involved in caching that need to be used when making caching decisions. They are, *Cache Replacement Policy* and *Cache Decision Policy*. In addition to these policies, the simulator has the possibility of using a real caching implementation or a mathematically modeled caching environment. These cache related aspects are explained below.

4.4.4.1 Cache Replacement Policy

The size of a cache is always limited due to caches being implemented on limited computer resources such as Random Access Memory (RAM) or disk storage. Due to these limitations, when contents need to be cached, a decision on what contents to be kept and what to be ejected must be made. This is called the *Cache Replacement Policy*. The simulator uses a replacement policy based on Least Recently Used (LRU). Work done in [PCL$^+$11], [RR11] and [MCG11] widely uses LRU in the context of evaluating CCN based networks.

With LRU, the content Chunks in a cache that have not been accessed recently are considered for rejection. The notion behind LRU is that if a Chunk which has not been accessed the longest is least likely to be accessed again in the near future.

4.4.4.2 Cache Decision Policy

Another policy employed in caches is the decision on what content Chunks are to be considered for caching. This is called the *Cache Decision Policy*. There are different policies used in CCN as shown in [RR11] such as Cache Always (ALWAYS) and Random with a Fixed Probability (FIX(P)). The most popular of these is ALWAYS [MCG11]. The simulator implements the ALWAYS mechanism where all Chunks received cached.

4.4.4.3 Real Caching

The LRU and the ALWAYS mechanisms are realized in the simulator using a linked list. Each of the entries hold the bytes of a Chunk with the prefix and segment number information. Once a Chunk is accessed, that Chunk is brought to the top of the list thereby pushing all the Chunks that are not accessed to the bottom of the list. When a new Chunk has to be cached, the lowest Chunk is ejected before the new Chunk is added to the top of the list.

One of the issues when using a real cache implementation is the time taken by a simulator to reach steady state. The steady state is reached when all caches are completely filled to represent the Zipf based popularity of content. Reaching steady state in simulated scenarios is dependent on the number of nodes in the network, content request rates of clients and sizes of caches. Therefore, the simulator also implements a mechanism to pre-fill caches with Chunks based on the Zipf model used. Filling of caches in this manner is done only once at the beginning of a simulation.

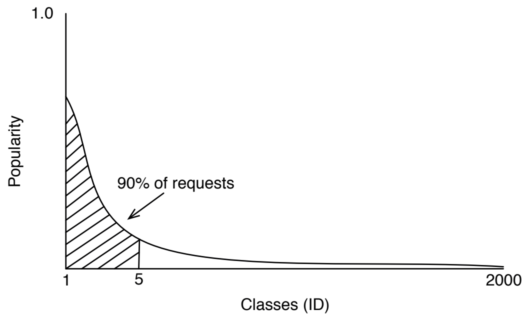


Figure 4.11: Content Coverage in a Cache

The pre-fill mechanism employs a routine that computes the content coverage percentage based on the Zipf parameters used and the cache size. Figure 4.11 shows a graphical view of the content covered by the cache given the parameters listed in Table 4.1. The figure shows that with these parameters, the cache is able to store up to (including) all contents of class 5, thereby providing a 90% content coverage.

Parameter	**Value**
Form of Zipf curve (α)	2.0
Number of Zipf classes	2000 classes
Number of content per class	10 contents
Number of Chunks per content	100 Chunks
Size of a cache in Chunks	5000 Chunks

Table 4.1: Parameters to Determine Content Coverage

4.4.4.4 Modeled Caching

The disadvantage of using real caching is the need to make RAM available in the simulator for each and every caching node to create and operate the caches. Since the RAM available is limited and certain scenarios may include hundreds of caching nodes, the simulator is also implemented with a caching mechanism that is modeled. In a modeled cache, a mathematical formula decides the probability of the availability of a requested Chunk in a cache.

The work done in [MCG11] identifies a caching model that combines the characteristics of LRU and a Zipf based popularity model for CCN. This model is represented by an equation (Equation 4.9) which shows the probability of a certain Chunk being not available in a cache.

$$p_k \sim e^{-\frac{\lambda}{m} q_k g x^\alpha} \tag{4.9}$$

where $1/g = \lambda c \omega^\alpha m^{\alpha-1} \Gamma\left(1 - \frac{1}{\alpha}\right)^\alpha$, x is the cache size in chunks, m is the number of contents in a class, λ is the arrival rate of Interests at the cache, q_k is the Zipf probability, ω is the size of a content in Chunks and α determines the form of the Zipf popularity model.

The simulator uses an extended version of this model (i.e., Equation 4.9) considering multi-path content retrievals by clients and the availability of multi-level caches. Chapter 6 provides the details of these extensions. The validations on using this model in the simulator is shown in Appendix A.

4.4.5 Pending Interest Table (PIT) Management

Entries are created or updated in the PIT when Interests are received and removed when corresponding Content Objects are received. As described in [JST$^+$09], a PIT may also be removed due to the expiration of the lifetime associated with the entry. This lifetime is computed when the PIT entry is created due to an Interest receipt or updated due to the subsequent receipt of an Interest requesting the same Chunk.

This PIT entry expiration mechanism operates solely on the basis of the information available at the CCN Layer (i.e., in *ccn_core*). Due to the Application Layer performing F&CC, there is a possibility that Interests resent by an application are not forwarded by the CCN Layer due to the PIT entry timer not expiring. F&CC in applications uses RTT to compute the expiration time of Interests (Section 4.3.4). When the RTO triggers, an application resends an Interest. When this Interest is received at the CCN Layer and if the PIT has an entry, the *ccn_core* simply updates the PIT without forwarding the Interest. But, when the PIT entry ultimately expires and is removed, the application is unaware of this information and therefore, would continue to wait until the RTO associated with the Interest expires. Therefore, due to the timers being unsynchronized, the application continues to wait without reacting which in turn degrades the application performance.

There are 2 possible solutions to handle this problem. The first is the implementation of a timer coordination mechanism between the Application Layer and the CCN Layer. But, the disadvantage is that, since the application is only executed at the client node, other intermediate CCN nodes such as routers require another mechanism.

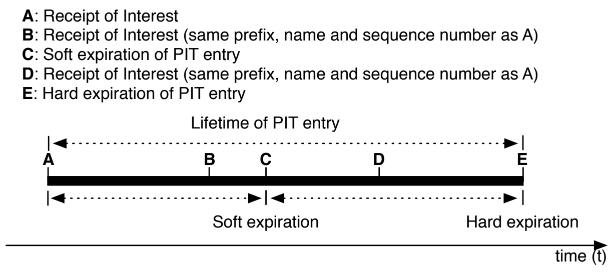

A: Receipt of Interest
B: Receipt of Interest (same prefix, name and sequence number as A)
C: Soft expiration of PIT entry
D: Receipt of Interest (same prefix, name and sequence number as A)
E: Hard expiration of PIT entry
Lifetime of PIT entry
A
B
C
D
E
Soft expiration
Hard expiration
time (t)

Figure 4.12: Operation of the Soft-Hard PIT Timeout Mechanism

The second solution, that is applicable to any CCN node and implemented at the CCN layer is the use of a dual timer mechanism which is called *Soft-Hard PIT Timeout Mechanism (SHPTM)*. Figure 4.12 shows the graphical view of the operation of SHPTM and Table 4.2 shows the events and actions performed by SHPTM.

Event	Actions
A: Receipt of Interest X	Create PIT entry Forward Interest
B: Receipt of a copy of Interest X	Update PIT entry
C: Soft expiration of PIT entry	Update PIT entry
D: Receipt of a copy of Interest X	Update PIT entry Forward Interest
E: Hard expiration of PIT entry	Remove PIT entry

Table 4.2: Events and Actions of the Soft-Hard PIT Timeout Mechanism

This mechanism maintains 2 sequential timeouts for every PIT entry. These timeouts, called *soft timeout* and *hard timeout* are created when the Interest is received for the first time. If a copy of the same Interest is received before the soft timeout, the PIT is updated and the Interest is not forwarded. When a copy of the same Interest is received after the soft timeout and before the hard timeout, the PIT is updated and also, the Interest is forwarded. When the hard timeout occurs, the PIT entry is removed from the PIT as described in [JST$^+$09].

The simulator is implemented with the SHPTM and the values used for the soft and hard timeout values are user configurable. The fundamental consideration that needs to be taken into account when deciding these timeout values is the balance that has to be maintained between not flooding the network with too many Interests and reducing the waiting time for content by an application. Both of these result in performance degradation for applications. When the soft timeout is longer, any Interest that arrives before the soft timeout would be forwarded resulting in flooding. A shorter soft timeout may result in the Interests resent by an application not being forwarded.

4.4.6 Path Usage

The simulator implements both path usage mechanisms described in Section 3.2.4, viz., Weighted Round Robin (WRR) and Leftover Capacity Utilization (LCU) for the OMP-IF forwarding strategy. In the case of WRR, the information on every path discovered for a content in the GFL, maintains a variable to hold the RTT computed from the last Interest-Content Object pair. This variable is used when computing the weight assigned to a path and then the Interests are forwarded through that path (i.e., Face) as described in Section 3.2.4.1.

In the case of LCU, the GFL maintains a variable for each content being requested that holds the path (i.e., Face) over which a Content Object arrived. When a subsequent Interest is required to be sent, the path in this variable is used to forward the Interest as described in Section 3.2.4.2.

4.4.7 Mobility Model

The scenarios identified to evaluate the mechanisms proposed in this thesis (Chapter 5) includes stationary networks and mobile networks. These evaluations consider CCN client nodes as having multiple network attachments connecting to different networks. Scenarios with stationary networks refer to networks where the CCN client nodes are permanently connected to their corresponding networks. Scenarios with mobile networks on the other hand refer to networks where CCN client nodes are mobile and therefore, have intermittent connectivity due to the movements.

The simulator implements a *Random Direction* based mobility model for CCN client nodes [CBD02]. In the mobility scenarios, there are 2 types of connectivity coverage supported by the network attachments of CCN clients. They are,

- **Wide Area Coverage**: Faces on a CCN client with *Wide Area Coverage* have continuous connectivity to a network in spite of the client being mobile. Therefore, in the scenarios considered, the Faces are considered to have permanent connectivity.
- **Local Area Coverage**: Faces with *Local Area Coverage* tend to have breakages in connectivity due to mobility of the CCN client.

Figure 4.13 shows a graphical view of a CCN mobile node with its movement based on a random direction and the coverage it experiences during mobility.

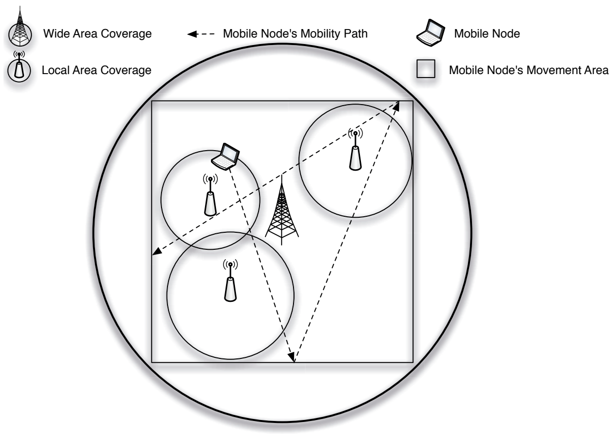


Figure 4.13: Random Direction based Mobility Model

The mobility model adopts a proactive update methodology of handling the operations related to disconnections and reconnections of Faces in CCN client nodes. This is similar to the operation of mobility with wireless network attachments where a threshold based mechanism is used to determine whether a Face is usable or unusable for communications. Once the change of the connectivity status is determined, this information is transferred across protocol layers to perform mobility related actions at those specific layers. A real world solution involving Wireless

Local Area Networking (WLAN) and Universal Mobile Telecommunication System (UMTS) network attachments has been developed and evaluated in [TUG$^+$08] and [FSU$^+$07]. All the mobility related updates are performed at the *ccn_core* as follows.

- **GFL Update**: When a Face gets disconnected, all entries in the GFL are updated by removing the Face from the list, if it exists (i.e., path was created over the Face). If no more Faces exist for the GFL entry, the compete GFL entry is removed to re-initiate a path discovery.
- **FIB Update**: If a disconnecting Face is associated with any of the FIB entries, those entries are updated by removing the Face and brought back when the Face is connected again.
- **Interest Resend**: When a Face gets disconnected, the Interests that were forwarded over the disconnecting Face are resent to make the corresponding Content Objects arrive over the Faces that continue to be connected.

There are 2 sets of configuration attributes used to define the parameters of the *Random Direction* model used. The first of these sets relate to the definition of the networks. They are,

- **Network Type**: Identifies whether a network attachment is of the type Wide Area Network (WAN) (i.e., has WAN coverage) or Local Area Network (LAN) (i.e., has LAN coverage). WAN is considered to have permanent connectivity while LAN has intermittent connectivity.
- **WAN Coverage Area**: Defines the dimensions of the rectangular area in which the mobile node moves and always has WAN connectivity [CBD02]. It is defined in terms of x,y coordinates and area.
- **LAN Coverage Area**: Defines the dimensions of the circular area in which LAN coverage (e.g., WLAN) is present. It is defined in terms of x,y coordinates and radius.

All dimensions above are defined in meters. The second set of configuration information defines the mobility characteristics of the mobile node. They are,

- **Random Number Generator (RNG) and Seed**: The mobile node follows a path based on a direction decided randomly. The RNG and the seed provides a set of uniformly distributed random numbers to decide the next traveling direction after the mobile node has reached a wall of the rectangular coverage area.

- **Pause Period**: Once a mobile node reaches a wall of the rectangular coverage area, it will pause for a random period. This random period is a uniformly distributed number representing seconds limited by an upper and a lower value.
- **Velocity**: The mobile node travels at a speed given by this attribute (in meters/second).

4.5 Adaptation Layer

CCN is overlaid on TCP/IP in the simulator. The Adaptation Layer is the interfacing layer between the CCN Layer and the TCP/IP Layer. CCN is a hop-by-hop communications model and the CCN model built on OPNET in this thesis uses the UDP as the underlying transport protocol to communicate hop-by-hop.

4.5.1 Adaptation Layer Process Model (*ccnal*)

The Adaptation Layer Process Model (*ccnal*) performs the tasks of converting CCN specific messages (i.e., Interests, Content Objects and Prefix Advertisements) received from the upper layers to UDP packets to be sent over the TCP/IP networks and vice versa. Figure 4.14 shows the FSM of *ccnal*. The FSM consists of the following states and transitions.

- **Init** and **Init2**: The Init and Init2 states are responsible for initializing variables and reading parameters associated with adaptation layer (e.g., UDP communication port number). The activities done by these 2 states have to be placed in 2 separate states due to Adaptation Layer Process Model (*ccnal*) requiring access to information at the IP layer (to obtain Face information). Due to the delays in IP layer initializations during simulations, *ccn_core* is required to access IP layer information with a delay. Therefore, the Init2 state is invoked with a delay of 100 milliseconds.
- **Wait**: The Wait state is where the FSM waits until an Adaptation Layer specific event occurs. A *CCN_CORE* is generated when an Interest, Content Object or Prefix Advertisement is received from the CCN Layer, resulting in a transition to the *RCVD_CCN* state. A *NETWORK* is generated when a UDP packet is received (encapsulating an Interest, Content Object or Prefix Advertisement) from the TCP/IP Layer, resulting in a transition to the *RCVD_NET* state. Finally, an *OPEN* is generated when an internal timer is

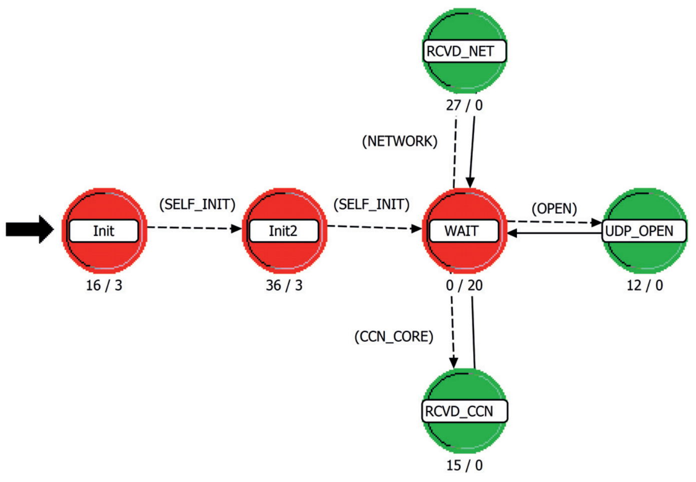


Figure 4.14: FSM of the CCN Adaptation Layer Process Model

triggered to setup the UDP communications, resulting in a transition to the *UDP_OPEN* state.

- **UDP_OPEN**: The UDP_OPEN state triggers functions to open the UDP port for communications through the network attachments connected to the CCN node. This is the port through which incoming and outgoing UDP packets are received and sent, respectively.

- **RCVD_CCN**: The RCVD_CCN state triggers functions to encapsulate the Interests, Content Objects or Prefix Advertisements received from the CCN Layer into UDP packets. These packets are then sent over the network attachment (i.e., Face) specified in the corresponding Interest, Content Object or Prefix Advertisement to the CCN node (i.e., next-hop).

- **RCVD_NET**: The RCVD_NET triggers a set of functions to decapsulate a UDP packet received from the TCP/IP Layer into the original Interest, Content Object or Prefix Advertisement that was encapsulated in that UDP packet. Once decapsulated, the Interest, Content Object or Prefix Advertisement is sent to the CCN Layer.

4.6 Transport and Network Layers

The CCN implementation in the OPNET simulator is overlaid on TCP/IP. OPNET provides an implementation of the complete TCP/IP protocol stack. The CCN model built in this thesis uses the Transport and the Network layers of TCP/IP to carry CCN messages.

4.6.1 Transport Layer (UDP)

As described in Section 4.5, the CCN Adaptation Layer encapsulates all CCN messages in UDP packets. The UDP port 3000 is used to communicate with other CCN nodes. The UDP payload carries the whole CCN message (i.e., Interest, Content Object or Prefix Advertisement). Due to the possibility of Content Objects carrying large payloads themselves, the TCP/IP Transport Layer is configured to use Jumbo UDP packets. At the IP Layer, these UDP packets are fragmented. The UDP Process Model (*udp*) handles the functionality related to processing UDP packet in OPNET (Figure 4.4).

4.6.2 Network Layer (IP)

The IP Layer of OPNET is set up to auto-configure Internet Protocol version 4 (IPv4) addresses for each of the network attachments that a CCN node has. Each network attachment (considered as a single CCN Face) is directly connected with another network attachment of another CCN node (see Figure 4.3) and they form 2 TCP/IP nodes belonging to a single IP sub network.

Communications between the nodes (i.e., 2 nodes) occur over broadcast communications, i.e., the UDP packets received from the transport layer (which encapsulates an Interest, Content Object or Prefix Advertisement) is broadcast to the network. The operation of propagating messages in CCN is based on multicast or broadcast. Therefore, the IP Layer adopts the same mechanism to propagate the UDP packets. The IP Encapsulation Process Model (*ip_encap*), IP Process Model (*ip*) and ARP Process Model (*arp*) handle the functionality related to the IP Layer in OPNET (Figure 4.4).

4.7 MAC Layer

The CCN model built in this thesis uses the Ethernet models of the OPNET simulator to handle the Link layer functionality. As shown in Figure 4.4, each network attachment is handled using a number of process models. They are,

- ARP Process Model (*arp*): Handles the functionality related to MAC to IPv4 address resolutions and vice versa through the Address Resolution Protocol (ARP).
- MAC Process Model (*mac*): Handles the MAC Layer related functionality such as handling queues.
- Ethernet Receiver Process Model (*eth_rx*) and Ethernet Transmitter Process Model (*eth_tx*): Handles the Ethernet packet receiving and transmitting related functionality of the communications.

The CCN model uses the 10BaseT, 100BaseT, 1000BaseX and 10Gbps_Ethernet modules of OPNET in the stationary as well as the mobile scenarios described in Chapter 5. In the mobility scenarios, these link capacities are reconfigured to represent Long Term Evolution (LTE) and WLAN network attachments. Details of the link capacities used in the scenarios are described in Section 5.2.5.

5 Performance Evaluation of Multi-path

The purpose of this chapter is to present the results and the analysis of the simulations performed using the mechanisms developed in this thesis. The evaluations consider two scenarios in large scale deployments. The performance of the mechanisms proposed in this thesis is compare against the performance of the *best-face* strategy proposed in [JST$^+$09], i.e., by Palo Alto Research Center (PARC) and extended by the NDN project [YAW$^+$12].

The sections in this chapter are organized as follows. The chapter begins by describing the scenarios considered in the evaluations, including the AT&T Inc. topology used to setup the simulated networks. The subsequent sections focus on the simulation parameters, the different simulation cases, the metrics identified for the evaluations and finally, the performance results and the analysis. The performance analysis section, in addition to the analysis of the mechanisms proposed in this thesis, includes an analysis of a set of issues associated with Content Centric Networking (CCN).

5.1 Considered Scenarios

The evaluations of the mechanisms proposed in this thesis are based on two scenarios. CCN is currently deployed only in testbeds. The scale of these deployments are small. The focus of this thesis is to evaluate the proposed mechanisms in a large scale setting and therefore, two probable large scale deployment scenarios are defined in this thesis. The first is the deployment of CCN in Fixed Network Operator (FNO) based stationary networks while the second relates to deployments in Mobile Network Operator (MNO) based networks. The following sections provide details of these scenarios.

5.1.1 Network Based Multi-path Scenario (Fixed Network Operator (FNO) based Deployments)

Fixed Network Operators (FNOs) are network connectivity providers to a group of subscribers. There are different kinds of FNOs based on the level at which they operate. *Tier 1* type of operators are the network operators that operate and maintain backbone high speed networks via interconnections that may also span over different geographical continents. The subscribers of such operators are the *Tier 2* type operators who sell their connectivity services to the *Tier 3* operators. *Tier 3* operators, referred to also as Internet Service Providers (ISPs), provide Internet connectivity services to subscribers such as homes and businesses.

ISPs typically subscribe to multiple *Tier 2* type operators to obtain connectivity to the Internet. Therefore, a single ISP may have multiple paths to reach the Internet. Further, an ISP may have to handle requests for a large number of content downloads due to a large customer base that an ISP may have. In such a situation, the routers of an ISP may become the bottlenecks. Therefore, an ISP may deploy multi-path mechanisms to avail the benefits of optimum resource utilization. This is referred to as network based multi-path in this thesis as the multi-path operations (i.e., multi-path discovery, utilization and maintenance) are done in the network as opposed to at the clients connected to an ISP.

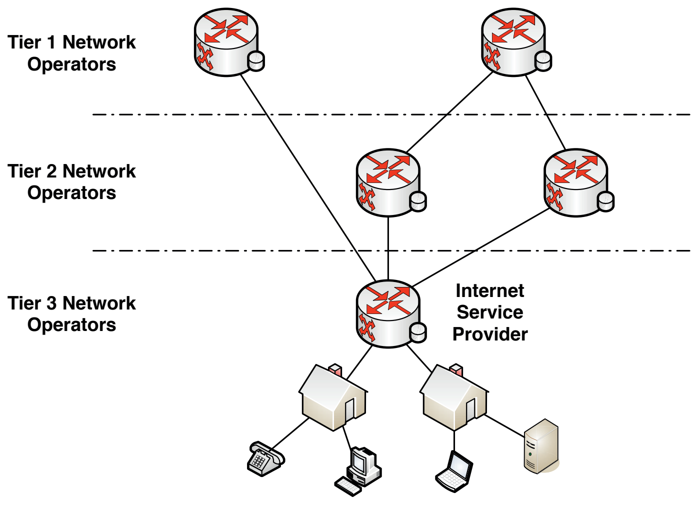


Figure 5.1: CCN Deployment on Stationary Networks operated by FNOs

Figure 5.1 shows a scenario of an ISP with multiple connections to the Internet (over *Tier 1* and *Tier 2* operators) and multiple subscribers. Considering that all these networks are operated on CCN based networks, the ISP is able to employ multi-path forwarding to manage the content download requests of its own subscribers.

Therefore, the purpose of this scenario is to evaluate the performance with network based multi-path.

5.1.2 Client Based Multi-path (Mobile Network Operator (MNO) based Deployments)

Mobile computing devices are proliferating the communications landscape due to the way people communicate all over the world. One of the main uses of mobile computing is the retrieval of content [GR11]. Therefore, CCN based networks are ideally suited to serve the needs of users in mobile networks operated by MNOs.

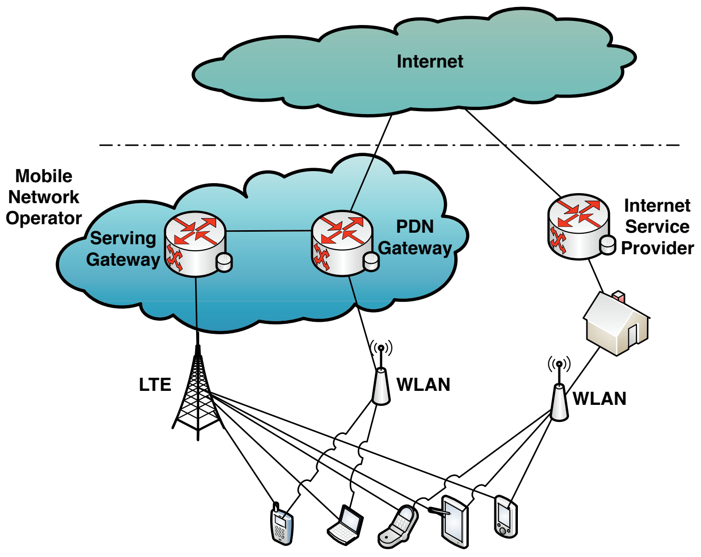


Figure 5.2: CCN Deployment in Mobile Networks

Figure 5.2 shows a deployment scenario with mobile devices equipped with multiple attachments connecting to different bearer technologies. Mobile devices in this scenario have multiple attachments that are either connected to a network of an MNO operating a 3rd Generation Partnership Project (3GPP) based network or an ISP. Due to availability of multiple attachments at the client side (mobile device), the multi-path mechanisms are deployed at the client side and the user is able to avail the benefits of optimum resource utilization.

Therefore, the purpose of this scenario is to evaluate the performance of the mechanisms proposed in this thesis in the context of a client based multi-path in mobile networks.

5.2 Simulation Environment

Evaluation of the multi-path mechanism proposed in this thesis considering the scenarios mentioned in Section 5.1 require the use of an appropriate network topology. The scale of nodes in such a network topology should be representative of networks operated by ISPs and MNOs. Therefore, without resorting to an artificially generated network topology, a real network topology is used to create these scenarios. The following sections describe the topology used and the different evaluation cases considered.

5.2.1 Network Topology

The authors of [HPSS03] have made available, the topology information of a 154-node Point of Presence (PoP) type setup hosted by the telecommunications operator AT&T Inc. in Extensible Markup Language (XML) encoding, as part of the research done into identifying realistic network topologies for simulations. This topology is used in the evaluations done in this thesis by converting these 154 nodes into CCN routers to form a flat CCN based network.

A CCN node deployed with the Server Application Process Model (*server_app*) is introduced as a content server. A number of CCN client nodes are introduced which are deployed with the Client Application Process Model (*client_app*). The links between the nodes have differing capacities. The core links have higher capacities while the links in the periphery have lower capacities. These range from 10Mbps to 10Gbps.

Figure 5.3 shows the FNO scenario in Section 5.1.1 setup in OPNET using the AT&T Inc. topology.

The scenario consists of 5 CCN routers located in the periphery of the network that act as the routers serving customers of an ISP. Each of these CCN routers serve 2 clients which request contents continuously. These CCN routers have multiple attachments to other CCN routers in the network.

The MNO scenario described in Section 5.1.2 is set up using the same network topology as shown in Figure 5.3 by considering the CCN client nodes as mobile nodes. The Faces attached to these CCN mobile nodes connect/disconnect to/from a set of CCN routers based on the mobility patterns used.

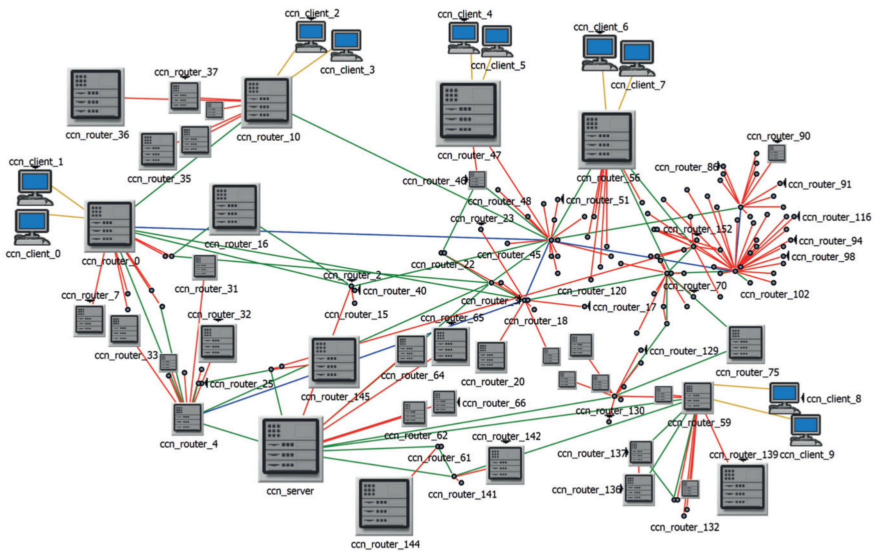


Figure 5.3: OPNET Scenario for Network based Multi-path with CCN

This scenario consist of 10 CCN mobile nodes which have differing number of multiple attachments (between 2 and 4) to some of the CCN routers in the network. In both scenarios, differing Background Traffic Load (BTL) levels, viz., 0 %, 30 % and 60 % of the link capacity are applied to emulate loaded links in networks. This load is generated as a constant bit rate on the link.

Each of the CCN routers in the network topology in both scenarios has differing number of network attachments. As indicated previously (Section 2.6), a CCN router performs 2 tasks, viz., forwarding and caching. Therefore, the CCN routers in the network topology which have more than one network attachment perform both of the tasks while the CCN routers with a single network attachment operates only as caches.

5.2.2 Popularity Model and Caching Configurations

The applications configured to execute on the CCN clients in both scenarios generate content requests based on the Zipf model and the naming convention described in Section 4.3.3. The flow and congestion control adopted by the applications are as described in Section 4.3.4. Table 5.1 shows the parameter values used in the simulator for the popularity model.

Parameter	Description	Value
M	Size of content catalog in files	20,000
K	Number of content classes	2,000
M/K	Number of contents per class	10
ω	Size of a content in Chunks	100
P	Size of a Chunk in bytes	10,000
α	Value characterizing the form of the Zipf model	2.0

Table 5.1: Popularity Model Parameters Used in Simulations

The values identified in Table 5.1 are based on the evaluations done in [RR11], [MCG11] and [CKR$^+$07]. [RR11] and [MCG11] evaluated the behavior of the Zipf popularity model in simulated CCN based environments. [CKR$^+$07] identified the best α value to use when characterizing the content popularity of You Tube.

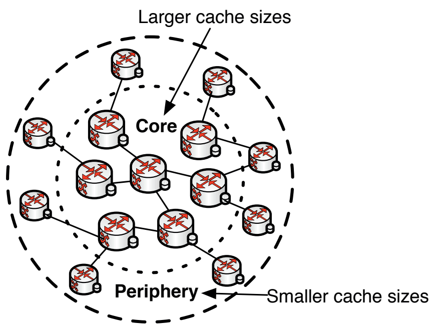


Figure 5.4: Configuration of Cache Categories

All routers in both scenarios are configured to cache content that flows through them. The caches use Least Recently Used (LRU) and Cache Always (ALWAYS) caching policies as described in Section 4.4.4. When considering the storage sizes of the caches and the processing powers of the servers that perform the caching in todays networks, an observation can be made as to how these servers are configured. Hypertext Transfer Protocol (HTTP) proxies are hosted on servers with limited capacities, close to the users (i.e., periphery) while Content Distribution Network (CDN) servers in the core of networks are configured with larger capacities to serve a large number of customers. To capture this observation, caches in the above scenarios are categorized into 2 groups as described below and as shown in Figure 5.4.

- Caches in the **Core**: The caches in the core are considered as having higher capacities and therefore are able to serve a larger proportion of content requests. Considering a request serving rate of about 90% in a cache, the size of the cache to hold the most popular Chunks is computed using the Cumulative Distribution Function (CDF) of the Zipf model (discussed in Section 4.3.3). Therefore, to serve 90% of the content requests, the caches must be configured to hold 5,000 Chunks (i.e., $x = 5,000$) which in turn results in the cache being able to cache up to (and including) content class 5 of the content catalog.
- Caches in the **Periphery**: The caches close to the client nodes (called the periphery) have relatively lesser capacities and therefore are configured to only serve about 75% of the content requests. The cache sizes in the periphery, computed using the CDF of the Zipf model are therefore set to 2,000 Chunks, covering up to (and including) content class 2 (i.e., $x = 2,000$).

These computations are based on the Zipf parameters listed in Table 5.1. The 90% coverage for the caches in the core of the network is based on the work done in [MCG11] while the 75% coverage value for the periphery is arbitrarily selected in this thesis with the intention of identifying a lower cache coverage.

Another aspect that shows varying performance is how the caches in the above scenarios are populated. There are 2 possibilities of cache populations considered in the simulations as described below.

- **Pre-filled Caches**: This refers to caches that are filled before simulations commence with content based on whether the cache belongs to the **Core** or **Periphery**. Thereby, the evaluations consider caches that have reached the steady state.
- **Empty Caches**: This refers to caches that have no content cached when the simulations begin. With empty caches, the evaluations focus on the behavior of performance during the period when the cache has still not reached the steady state.

5.2.3 Flow and Congestion Control (F&CC) Configurations

The application employ F&CC as described in Section 4.3.4 to control the flow of the generated Interests to adapt to network conditions. F&CC uses a set of state

Variable	Description	Value
cwnd	Congestion window value in bytes	10,000
ssthresh	Slow start threshold value in bytes	65,535
RTO	Re-transmission Timeout (RTO) value in seconds	3.0

Table 5.2: Initial Values of F&CC Variables

variables to perform its activities and these are initialized as shown in Table 5.2 at the start of each content download.

5.2.4 Pending Interest Table (PIT) Timeout Configurations

The PIT uses a dual timeout mechanism as stated in Section 4.4.5 to avoid timeout clashes with the F&CC implemented by the applications in the simulator (Section 4.3.4). Due to the issues that the Soft-Hard PIT Timeout Mechanism (SHPTM) attempts to avoid, the *soft timeout* should be larger than the Round Trip Time (RTT) of an Interest-Content Object pair. At the same time, the *soft timeout* should also be less than the minimum RTO used by F&CC. The RTO computation in F&CC is based on [PACS11] and therefore, uses 1 second minimum specified in [PACS11]. Table 5.3 shows the SHPTM timeout values used in simulations.

Parameter	Value
Soft Timeout	0.5 seconds
Hard Timeout	1.5 seconds

Table 5.3: PIT Timeout Values Used in Simulations

5.2.5 Link Capacities

The FNO and MNO scenarios use differing link capacities to represent the different link technologies (i.e., network interfaces) used in the scenarios. In the case of FNO scenario, the standard Ethernet modules of OPNET are used for the CCN Routers and the CCN Clients. With the MNO scenario, the same Ethernet nodules are used with link capacities adjusted to represent Long Term Evolution (LTE) (i.e., Wide Area Network (WAN)) network interfaces and Wireless Local Area Networking (WLAN) network interfaces. Table 5.4 shows the link capacities configured in the scenarios. *Path Discoverer* is the CCN Node that initiates the path discovery based on the forwarding strategy deployed (i.e., Best-face or On-demand Multi-path - Interest Forwarding (OMP-IF)). In the case of the FNO scenario, the

CCN Router performs the path discovery while in the MNO scenario, the CCN Clients perform the path discovery. Figure 5.4 shows a graphical view of the locations of the CCN Routers (i.e., Core and Periphery) referenced in the Table 5.4.

Scenario	**Link Capacities**			
	CCN Routers (Core and Periphery)	**Path Discoverer**		
		CCN Router	**CCN Client**	
		LAN	**WAN**	**WLAN**
FNO	10 Gbps, 1 Gbps, 100 Mbps	10 Mbps		
MNO	10 Gbps, 1 Gbps, 100 Mbps		7 Mbps	9 Mbps

Table 5.4: Link Capacities Used in Scenarios

5.2.6 Mobility Parameters

The MNO based scenario consists of mobile nodes that are mobile and therefore, have breakages in their connectivity. Mobility of nodes in the simulator is handled as described in Section 4.4.7. Each of the mobile nodes in this scenario has between 2 and 4 network attachments (Faces) which are randomly assigned to the nodes. One network attachment is always considered as a WAN type attachment while the others are considered as Local Area Network (LAN) type attachments. The mobile nodes use the *Random Direction* based mobility model.

Parameter	**Value**	
CCN client velocity	1.4 [m/sec]	
	Lower Bound	Upper Bound
Pause time	5 [sec]	10 [sec]
WAN coverage	Begin x and y coordinates [m]	Area [m^2]
Face 1	0,0	500 x 500
LAN coverage	Center x and y coordinates [m]	Radius [m]
Face 2	150,110	100
Face 3	390,300	100
Face 4	290,390	100

Table 5.5: Mobility Parameters used by CCN Client Nodes

Every mobile node uses a common set of parameters defined in the mobility configuration. Table 5.5 shows a list of these parameters. Figure 5.5 shows a proportionally drawn coverage map of the MNO scenario based on the data given in Table 5.5.

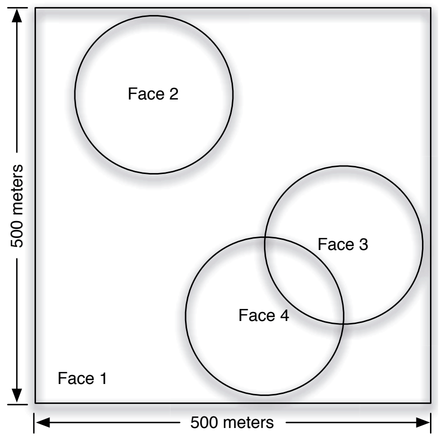


Figure 5.5: Coverage Map of MNO Scenario

5.2.7 Simulation Durations and Seeds

The simulations are executed for 10,800 seconds in each of the scenarios and the cases. This duration is determined after considering the time taken by some of the macro scale metrics (described in Section 5.4.1) to reach a steady state. At the beginning of every simulation, time is allocated for the *server_app* to initiate Forwarding Information Base (FIB) populations as described in Section 4.4.2. The CCN client nodes commence the continuous content downloads after 300 seconds. The download commence time is determined after considering the time required for initializations performed by each CCN client.

A single simulation case is executed 10 times with differing seed values to obtain the confidence intervals according to the *Student-t* distribution. The Appendix B lists the mean values and their 95% confidence interval values for the results shown in this chapter.

5.3 Simulation Cases

The FNO and the MNO scenarios described above are the base on which all simulations are performed. Each of these scenarios has multiple possibilities based on the different configuration information. These are referred to as the *Simulation Cases* in this thesis. Following is a list of the tags separated by category, used to name the different combinations of configuration information which are then combined to form Simulation Cases.

Scenario

- **FNO**: Simulations are done using the FNO based scenario.
- **MNO**: Simulations are done using the MNO based scenario.

Forwarding Strategy

- **BF**: All CCN nodes in the network use the best-face forwarding strategy to forward Interests.
- **MP**: All CCN nodes in the network use the OMP-IF strategy identified in this thesis to forward Interests.

Caching Configuration

- **WCS**: All CCN routers in the network are enabled with caching and therefore all Chunks traversing a router are cached.
- **NCS**: No CCN router in the network is enabled with caching and therefore all Interests reach the server hosting the content.
- **RCS**: When caching is enabled (i.e., WCS), the real LRU caching is used in each of the CCN routers.
- **MCS**: When caching is enabled (i.e., WCS), the modeled LRU caching is used in each of the CCN routers.
- **PCS**: When real caching is enabled (i.e., WCS and RCS), all CCN routers are pre-filled with Chunks based on the Zipf popularity model, at the beginning of the simulations.
- **ECS**: When real caching is enabled (i.e., WCS and RCS), all CCN routers start with empty caches at the beginning of the simulations, letting the caches be filled through CCN client content downloads.

Interest Distribution Mechanism

- **LCU**: When the OMP-IF forwarding strategy is used, the distribution of Interests to multi-path is done based on the Leftover Capacity Utilization (LCU) mechanism.

- **WRR**: When the OMP-IF forwarding strategy is used, the distribution of Interests to multi-path is done based on the Weighted Round Robin (WRR) mechanism.

Background Traffic

- *nn***BTL**: Links are loaded with differing levels of background traffic, i.e., BTL, with *nn* referring to the % of the artificial Constant Bit Rate (CBR) load placed on a link as a % of the link capacity (e.g., 30BTL on a 100Base-T refers to a BTL of 30% of 100 Mbps, i.e., 30 Mbps).

Popularity Renewal

- **C***nn***PR**: Partial renewal of the popularity model by removing a content (i.e., all Chunks of a content) from all the caches in the network with *nn* referring to the Zipf class from which the content is removed.

Based on the above information, the name of an example Simulation Case where the FNO scenario is configured to operate with the OMP-IF strategy, pre-filled caches, WRR based traffic distribution, modeled caching and links loaded with 30% BTL is as follows.

FNO-MP-WCS-RCS-PCS-WRR-30BTL

5.4 Evaluation Metrics

The evaluation of the mechanisms proposed in this thesis can be performed at the network level or at the CCN node level. The network level evaluations (*Macro Scale*) focus on evaluating the combined performance of multiple CCN nodes deployed with the mechanisms proposed in this thesis. An example is how well the load induced by the content downloads is balanced throughout the network. The node level evaluations (*Micro Scale*) focus on the performance of individual CCN nodes. An example is the *cwnd* of a CCN client node which shows how well the Content Objects are being downloaded (e.g., pace of a download).

The simulator is implemented with evaluation metrics that focus on both scales. But, the evaluations of this thesis focus mainly on the Macro Scale metrics to determine the benefits or disadvantages of using the proposed mechanisms in large scale networks (i.e., FNO and MNO scenarios). Therefore, the majority of the metrics used in the evaluations are Macro Scale metrics. A few Micro Scale metrics are also used when CCN node level performance details are required in the analysis. The following sections describe the metrics used in the evaluations.

5.4.1 Macro Scale Metrics

Macro Scale metrics are computed based on aggregating the statistical values produced by all the individual CCN nodes in an evaluated scenario. The implementation uses the globally scoped statistics in OPNET process models to accumulate these values during a simulation run. These metrics are computed as cumulative averages.

5.4.1.1 Average Cache Hit Ratio

A cache hit ratio refers to the ratio between the times that a cache was able to serve a Chunk and all the requests that were received at the cache. It is a measure that indicates how well a cache is serving Chunks. This metric, the Average Cache Hit Ratio refers to the combined hit ratio of all the hit ratios of individual cache in a network irrespective of the content class. The Average Cache Hit Ratio is computed as follows.

$$\text{Average Cache Hit Ratio} = \frac{\sum_{i=1}^{N} C_{H,i}}{\left(\sum_{i=1}^{N} C_{H,i}\right) + \left(\sum_{i=1}^{N} C_{M,i}\right)}$$

where N refers to the number of caches in the network (excluding the server), $C_{H,i}$ refers to the number of times an Interest was able to be served at cache i (i.e., number of hits) and $C_{M,i}$ refers to the number of times an Interest was not able to be served at cache i (number of misses). The Average Cache Hit Ratio is computed at the CCN Layer where the Content Store (CS) is maintained.

5.4.1.2 Average Content Download Time

Average Content Download Time refers to the average time taken by a complete content (i.e., a file) to be downloaded. A content consists of a number of Chunks which are requested using Interests and received as Content Objects. This metric is an average computed over all the contents downloaded by all the CCN client nodes. The Average Content Download Time is computed as follows.

$$\text{Average Content Download Time} = \frac{1}{F}\sum_{n=1}^{F} T_f$$

where F is the number of all contents downloaded by all CCN client nodes during the simulation and T_f is the download time of the content f ($f = 1, 2, ..., F$).

5.4.1.3 Average Content Object Delay

Average Content Object Delay refers to the average time taken by a Content Object to reach the CCN client node that requested the Content Object, from the CCN node that sent the Content Object (i.e., originator). A Content Object may originate from either a CCN router that has cached the Content Object or a CCN server that executes the *server_app*. The Average Content Object Delay is computed as follows.

$$\text{Average Content Object Delay} = \frac{1}{N}\sum_{n=1}^{N} D_n$$

where N is the number of all Content Objects downloaded by all CCN client nodes during the simulation and D_n is the time taken for the Content Object n to reach the CCN client node, from the originator ($n = 1, 2, ..., N$).

5.4.1.4 Average Content Object Hop Count

Average Content Object Hop Count refers to the average number of hops that a Content Object travelled to reach the CCN client node that requested the Content Object. The Average Content Object Hop Count is computed as follows.

$$\text{Average Content Object Hop Count} = \frac{1}{N}\sum_{n=1}^{N} H_n$$

where N is the number of all the Content Objects downloaded by all CCN client nodes during the simulation and H_n is the number of hops travelled by the Content Object n to reach the CCN client node, from the originator ($n = 1, 2, 3, ..., N$). A Content Object may originate from either a CCN router that has cached the Content Object or a CCN server that executes the *server_app*.

5.4.1.5 Load Balancing Measure

Load Balancing Measure refers to the metric that is used to determine how well the load in a network is spread across the network. [YCHP08] proposes the use of the Coefficient of Variation (CoV) as an indicator to measure the distribution of traffic. In this thesis, the CoV of Content Objects that are forwarded by a CCN router to all the Content Objects forwarded by all the CCN routers in the network is used as the Load Balancing Measure. It is computed in the following manner.

$$\begin{aligned}\text{Load Balancing Measure} &= CoV(f) \\ &= \frac{stdev[f(v)]}{E[f(v)]}\end{aligned}$$

where $f(v)$ is the function that returns the number of Content Objects forwarded at a CCN router node v and $v = 1, 2, 3, ..., V$. V is the total number of CCN routers in the network. Smaller values of the Load Balancing Measure is an indication of the load being well spread over the network while larger values are an indication of unevenly spread loads.

5.4.1.6 Average Retrieval Ratio

Average Retrieval Ratio refers to the metric that indicates how well the Content Objects are retrieved by Interests. In CCN an Interest brings back only one Content Object at most. But, due to a number of reasons such as congestion in the network and mobility of CCN client nodes, there might be instances where more than one Interest is issued to retrieve a Content Object. Average Retrieval Ratio is computed in the following manner.

$$\text{Average Retrieval Ratio} = \frac{\sum_{i=1}^{N} f(F_{C,i})}{\sum_{i=1}^{N} f(F_{I,i})}$$

where N is the number of CCN client nodes in the network, $f(F_{I,i})$ is the function that returns the total Interests by the CCN client node i and $f(F_{C,i})$ is the function that returns the total Content Objects received by the CCN client node i. A value equal to 1.0 indicates an optimum performance of all CCN client nodes while a value less than 1.0 indicates inefficiencies in the performance.

5.4.1.7 Interest Flow Efficiency

Interest Flow Efficiency refers to the metric that shows how the Interests that were received by a CCN node were forwarded or what possibilities the CCN node had to forward these Interests. This metric is mainly used to show the efficiency of the OMP-IF strategy and represented as a percentage of all the Interests that traversed a CCN node. It consists of the following information.

- Interests that were broadcast to all Faces during path discovery (i.e., *Probe* Interests).
- Interests that had only a choice of selecting one path because the path discovery had only created one path in the Good Face List (GFL).
- Interests had the choice of 2 paths to select from (due to GFL having multiple Face entries).

The computation of this metric is done based on the above information, collected from different CCN node types depending on the scenario. In the case of the FNO scenario, the information is collected from the CCN router nodes that perform the multi-path discovery, use and maintenance. In the case of the MNO scenario, the information is collected from the CCN client nodes.

5.4.2 Micro Scale Metrics

Micro Scale metrics are computed per CCN node. The implementation uses locally scoped statistics in OPNET process models to accumulate these values during a simulation run. These metrics are computed as cumulative averages.

5.4.2.1 *cwnd*

The cwnd metric refers to the congestion window maintained at the CCN client nodes by F&CC. It is set to an initial value at the beginning of a content download and is adjusted during the content download.

5.5 Simulation Results and Analysis

This section presents the simulation results and the analysis of the performance of the multi-path mechanisms proposed in this thesis. This section consists of two sub sections, one focusing on the FNO scenario while the other is focussing on the MNO scenario. Within each of these scenarios, the fundamental performance comparison is between how the forwarding strategy proposed in [JST$^+$09] (i.e., the original creators of CCN) fares against the multi-path mechanisms proposed in this thesis.

[JST$^+$09] proposes two forwarding strategies, viz., best-face strategy and the standard strategy. The standard strategy is proposed only as a means of explaining the operation of CCN, but the performance results in [JST$^+$09] are based on the best-face strategy. The authors of [RR11] have also evaluated the standard strategy in a multi-path context and have described the inferior performance in this strategy. Therefore, in all the performance comparisons in this thesis, the best-face strategy is used.

Each of the results presented in the following sections are performed using the parameters listed in the tables given in the previous sections and based on the cases formed using the tags listed in Section 5.3. Table 5.6 shows the applicable parameter tables relevant for the scenarios.

Scenario	**Parameter Tables**
FNO	Table 5.1 (Page 86), Table 5.2 (Page 88), Table 5.3 (Page 88) and Table 5.4 (Page 89)
MNO	Table 5.1 (Page 86), Table 5.2 (Page 88), Table 5.3 (Page 88), Table 5.4 (Page 89) and Table 5.5 (Page 89)

Table 5.6: Parameter Tables used in the Scenarios

5.5.1 Empty Caches vs. Pre-filled Caches

CCN client nodes continuously download contents during the simulations. The Interests that are generated to obtain each of the Content Objects are either served by a CCN router (i.e., cache) along the way or by the CCN server node which hosts the contents. At the beginning of a simulation, the caches are unable to serve any Interest with the corresponding Content Objects as the caches are empty. Over time, due to the downloads initiated by the CCN client nodes, the caches fill up.

When all the caches in a network are filled up, the Chunks that are in the caches represent the Zipf popularity model as the requests for contents followed the Zipf popularity pattern. This is considered as the steady state of a cache.

When considering simulations of large scale topologies such as the FNO scenario, the time taken to reach the steady state has been observed in the simulations to take longer as a large number of content downloads are required to fill the caches. Figure 5.6 shows the Hit Ratio comparison between a topology where

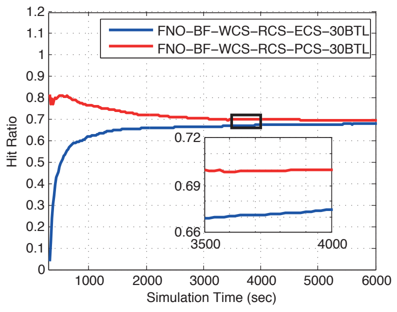
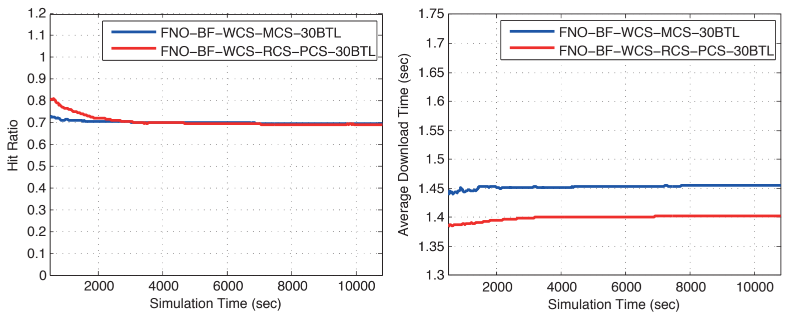


Figure 5.6: Comparison of Performance with Empty and Pre-filled Caches

caches are empty and where the caches are pre-filled. The scenario where the caches are not pre-filled shows that the Hit Ratio takes time to reach the steady state of around 70% while the scenario with the pre-filled caches reach the steady state much earlier. The reason for pre-filled caches not reaching the steady state right from the start is due to the effects of Zipf based content requests. Since the Zip based requests may result in requests for un-cached or cached content, the initial hit ratio values may be lower or higher, respectively. This lower or higher hit ratios will last until these initial fluctuations smooth out. Therefore, the rest of the simulations done in this section focus on pre-filled caches when considering real caches.

5.5.2 Modeled Caching with Flow and Congestion Control (F&CC)

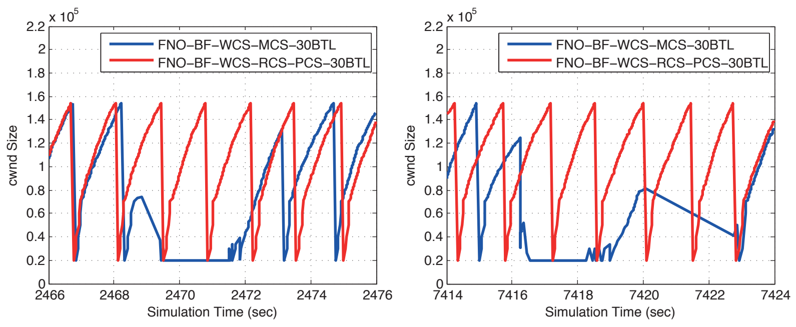
Modeled caching is an alternative to using real caching when simulating CCN based networks. The LRU based caching model implemented in the simulator (detailed in Section 4.4.4.4, Chapter 6 and Appendix A) uses a mathematical formulation to determine the availability of Chunks in a cache together with a random number. The real LRU cache implements the functionality with the appropriate storage to perform the get/set operations on storing and retrieving the Chunks. Figure 5.7(a) shows the combined performance of the hit ratio on all the caches with modeled and real LRU caching in the FNO case using the best-face forwarding strategy proposed by [JST$^+$09]. The curves show that both caching implementations closely match in handling hits and misses of requests for Chunks. On the other hand, Figure 5.7(b) shows a mismatch of performance in relation to the average download time of all the CCN client nodes.



(a) Cache Hit Ratio (b) Average Download Time

Figure 5.7: Comparison of Performance with Modeled and Simulated Caching

When evaluating the application performance of one single client, it was found that the performance of the congestion window (*cwnd*) of F&CC differed in their behaviors. Figure 5.8 shows two snapshots of *cwnd* performance, where the modeled caching case has sudden drops in the *cwnd*. These drops are due to RTO expirations. The reason for RTO expirations is due to some Interests requiring to travel a longer distance "*suddenly*".



(a) First Snapshot of *cwnd* Performance (b) Second Snapshot of *cwnd* Performance

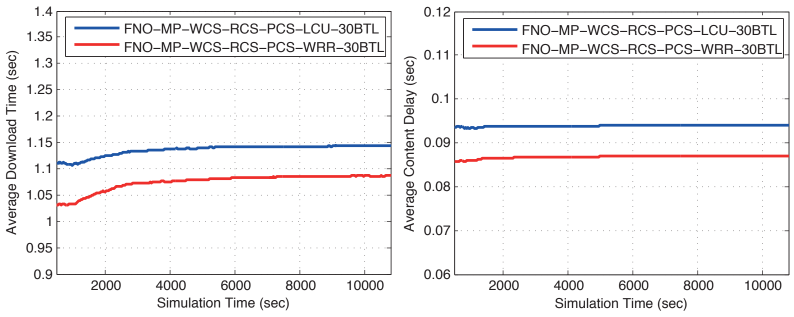
Figure 5.8: Congestion Window (*cwnd*) Performance of a Single CCN Client Node

In the real caching case, all Chunks of a requested content are likely to be present in a cache if the first Chunk is also available in that particular cache. Therefore, the RTO calculated by a client is based on the RTT to this cache. This may vary over time but generally sticks to a similar value. Therefore, RTO based expirations are highly unlikely. On the other hand, due to the use of random numbers to determine the existence of a Chunk in a cache in the modeled caching case, there is a likeliness that a Chunk of a content may "*suddenly*" be unavailable while the other Chunks before were available. This results in RTO expirations and hence the drop in the *cwnd*, affecting the performance of the downloads.

Concluding, though modeled caching generally performs as a real cache would do in terms of the hits and misses (see discussion in Appendix A), it has the above mentioned performance drawback when used in an environment where F&CC is used by CCN applications. Therefore, the rest of the performance evaluations in this chapter are configured to use real caching in the simulator.

5.5.3 Path Utilization

This thesis proposes two possible mechanisms of utilizing multiple paths with the OMP-IF forwarding strategy (Section 3.2.4). Utilization of paths refers to the way in which the Faces in GFL are identified to forward Interests. The WRR mechanism distributes the Interests based on a set of weights assigned to each Face (i.e., path). These weights are computed based on the delays associated with each Face. The LCU mechanism selects the Face through which the last Content Object was received to send Interests. Both of these mechanisms are intuitive in that, they automatically adjust to current network conditions.

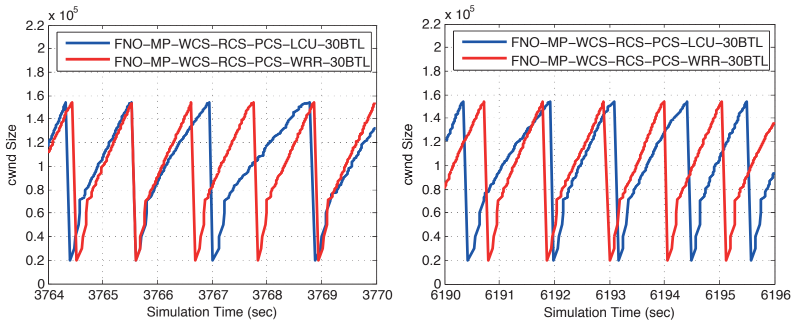


(a) Average Download Time (b) Average Content Object Delay

Figure 5.9: Comparison of Performance with LCU and WRR

With WRR, the implicit assumption is that a path with smaller delay is better than a path with a higher delay. Similarly, with LCU, a path that returns a Content

Object is better suited to send the next Interest. Figures 5.9 and 5.10 show a performance comparison of using LCU and WRR with the FNO based scenario.



(a) First Snapshot of *cwnd* Performance (b) Second Snapshot of *cwnd* Performance

Figure 5.10: Congestion Window (*cwnd*) Performance of a Single CCN Client Node

Figure 5.9(a) shows the progress of the average download time of all the CCN client nodes, continuously downloading content. This figure shows the OMP-IF strategy performing better when using WRR compared to LCU. Similarly, the average Content Object download time shown in Figure 5.9(b), also shows that WRR performs better.

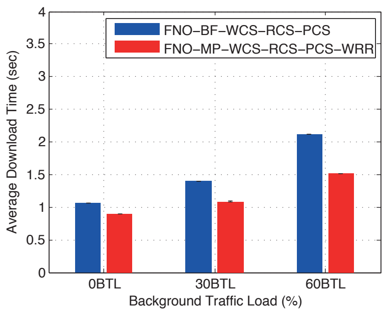
F&CC in CCN applications controls the speed at which a content is downloaded. One of the key control parameters in F&CC, the *cwnd* is influenced by the RTT computed using Interest-Content Object pairs. When the RTT increases, the growth of *cwnd* slows down and vice versa, when the RTT decreases. The increase of RTT occurs due to congestion or losses in the network. As Figure 5.9(b) shows, the higher average Content Object delay seen with LCU means that the RTT with LCU is higher than with WRR. Therefore, due to the larger RTT with LCU, the time taken to download a content is longer (i.e., average download times are higher). Figures 5.10(a) and 5.10(b) are snapshots of the progress of *cwnd* with LCU and WRR. The *cwnd* of WRR shows a steeper rise (i.e., faster growth) compared to the *cwnd* of LCU.

The conclusion from these observations is that, LCU results in more congestion in the network rather than WRR. The reason for the increased congestion is in the way that LCU operates. Even though we assume that a path may be able to carry more and more Interests when that path is able to bring back Content Objects, in reality this may not be true. Even when there is congestion, a path may be able to bring back a few Content Objects and due to the way LCU works, we send Interests over a congested path thereby increasing the congestion further.

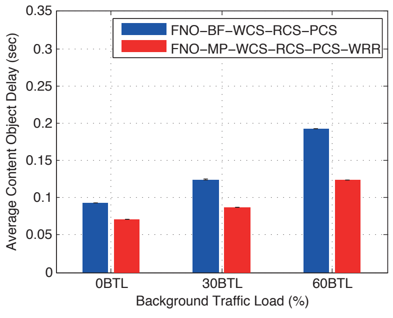
Therefore, due to the poor performance of the LCU mechanism, the rest of the performance evaluations use the WRR mechanism instead of LCU.

5.5.4 Performance in Fixed Network Operator (FNO) based Networks

This section focusses on the performance of MNO based networks when deployed with the multi-path mechanisms proposed in this thesis, compared to the mechanisms proposed in [JST+09]. There are two main characteristics in FNO based networks (in relation to MNO based networks). The first of these is that, all nodes are stationary and therefore connectivity to networks are permanent. The second characteristic is that, the CCN routers of an FNO perform multi-path operations instead of the CCN clients.



(a) Average Download Time



(b) Average Content Object Delay

Figure 5.11: Performance of Download Time and Content Object Delay in FNO Scenario

Figure 5.11(a) shows the performance of Content Download Time under differing BTL levels. The OMP-IF strategy performs better in all the BTL levels considered. The main reason for the better performance shown by OMP-IF strategy is the effect of bandwidth aggregation. With multi-path, the CCN client node distributes the Interests and thereby forces the Content Objects also to arrive over those paths. As more Content Objects arrive at any given time, compared to the best-face strategy, the F&CC would increase the *cwnd* at a greater pace.

In comparison to the OMP-IF strategy, the best-face strategy would identify a path initially and starts to send all Interests over this path. Though this path was the best in terms of delay at the time of the discovery, utilization results in the path being congested. As a result of this, the F&CC maintains a relatively smaller pace of *cwnd* increase to adapt to congestion. Therefore, the effect of this behavior is the reduced download time.

Figure 5.11(b), which shows the Average Content Object Delay, confirms the above observations. With the best-face strategy, the Content Objects arrive slower than when using the OMP-IF strategy. This is due to congestion that is experienced by a content download. This delay results in F&CC slowing down the *cwnd* growth.

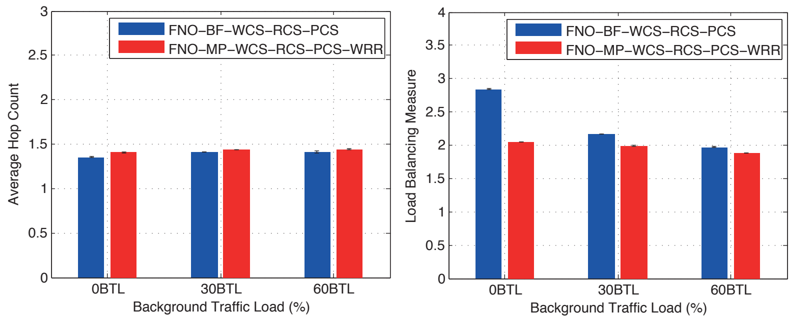
A further observation is the widening differences of performance of the 2 forwarding strategies when the BTL increases in the network. The more BTL that is present, the larger the Download Time and Content Delay becomes. But, due to the increase of the congestion along the selected path by the best-face strategy, the increase of delay is not proportional to the increase of the BTL. On the other hand, the OMP-IF strategy adapts to the conditions when using the paths. When a path is congested, this is reflected in the delay of an Interest-Content Object pair sent over that path and therefore, the OMP-IF strategy adjusts the distribution ratio assigning a lower weight to the congested path.

FNO: Average Download Time (sec)					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
1.062790 $\pm$0.000819	0.900476 $\pm$0.001984	1.401598 $\pm$0.001028	1.082871 $\pm$0.002840	2.113800 $\pm$0.001140	1.511923 $\pm$0.002984

Table 5.7: Performance of Download Time in FNO Scenario - Numerical Results

Table 5.7 shows the numerical results of the Average Download Time shown in Figure 5.11. It shows the sample mean and the margin of error values. Statistical estimations are performed using the Independent Replications methodology and the Confidence Interval (CI) is computed using the Student's t-distribution. The margin of error values are computed for a 95% confidence level. The details of the computations together with the numerical results of the subsequent figures are provided in the Appendix B.

Figure 5.12(a) shows the Hop Counts, i.e., the distance to the CCN router/cache or server that sent a Content Objects requested by a CCN client node, in terms of hops travelled. The best-face strategy shows a slightly better performance under all BTL levels. The reason for this better performance is as follows. In the case of the best-face strategy, the *Probes* sent at the time of path discovery, discovers a path that has the lowest delay due to the path being not congested at the beginning of a content download. The lowest delay path is usually the closest CCN router/cache



(a) Average Hop Count (b) Load Balancing Measure

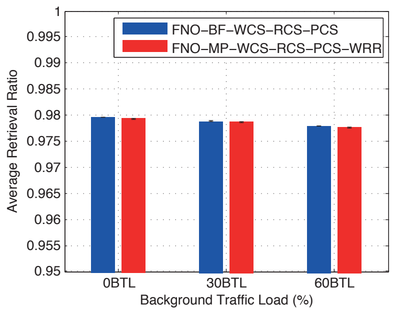
Figure 5.12: Performance of Hop Count and Load Balancing in FNO Scenario

or server. On the other hand, with the OMP-IF strategy, the first path used is identified in the same manner. But due to the discovery of paths based on node disjointness, the second path may not be the closest in terms of hops. Therefore, the averaging out of the hop counts results in the OMP-IF strategy having a higher hop count than the best-face strategy.

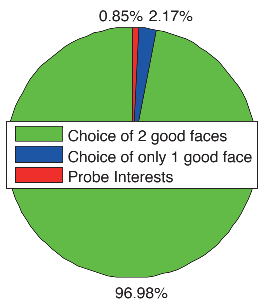
Figure 5.12(b) shows the Load Balancing Measure computed as described in Section 5.4.1.5. The lower the value the better the distribution of the load throughout the network. The OMP-IF strategy performs better under all BTL levels compared to best-face strategy. But, as the BTL level increases, the best-face strategy comes close to the performance levels shown with OMP-IF strategy. The reason for this is as follows. OMP-IF generally performs the best at any BTL level as the the paths discovered are node disjoint. As the load in the network increases, the best-face strategy too discovers paths with less delay thereby distributing the traffic to unused paths of the network.

Figure 5.13(a) shows the Retrieval Ratio. The Retrieval Ratio computes the ratio of Content Objects received to the Interests sent to fetch those Content Objects. In CCN, every Interest, at most, fetches only one Content Object [JST$^+$09]. But due to the following reasons, there could be more Interests sent than the number of Content Objects which were retrieved.

1. Commencing Path Discovery: The replication of Interests done by the forwarding strategies to discover paths at the beginning of a content download.
2. Subsequent Path Discovery: While a content is being downloaded, due to a break in all discovered paths, the forwarding strategies may perform a re-



(a) Average Retrieval Ratio



(b) Interest Flow Efficiency

Figure 5.13: Performance of Retrieval Ratio and Interest Flow Efficiency in FNO Scenario

discovery of paths through Interest replications.

3. Interest Expiration: F&CC may resend an Interest due to an RTO expiration.

Both, best-face and OMP-IF forwarding strategies employ Interest replication to discover (or re-discover) paths. Therefore, as seen from Figure 5.13(a), the Retrieval Ratio will never reach the 1.0 value due to more Interests being sent than the received Content Objects. The reason for not reaching the 1.0 value therefore is due to the influence of the above 3 reasons which adds up to about 2% in all the Retrieval Ratios shown.

A further observation is the similarity of the values of all the Retrieval Ratios. Even though, the lower download times with best-face strategy means a lower number of Interests sent compared to the OMP-IF strategy, the Retrieval Ratios will have a close match due to the above 3 reasons being equally applicable for both strategies.

Finally, another observation in Figure 5.13(a) is the slight reduction in the Retrieval Ratio values when the BTL increases. In an unloaded network (i.e., no BTL), the likely cause for the replication of Interests is the reason 1 mentioned above. But, with the increase of BTL, reasons 2 and 3 result in generating additional Interests.

Figure 5.13(b), the Interest Flow Efficiency shows a breakup of the way the Interests were forwarded by the OMP-IF strategy when no BTL was applied in the network. The 0.85% portion of the pie chart consists of the reasons 1 and 2. The 2.17% portion is where an Interest when received at the router that performs multi-path has only a choice of one Face to select as the path discovery is on-going.

96.98% of the time, the Interests were able to be forwarded after the multiple paths are discovered. This pie chart shows the efficiency of OMP-IF and even though the best-face may have a similar percentage of *Probe* Interests to discover the path to use, OMP-IF is superior due to the performance gains in terms of reduced download times (see Figure 5.11(a)).

5.5.4.1 Download Analysis - Single Content

A retrieval of a single content by a CCN application involves a number of activities that influences the final download time of the content. They are as follows.

- **Path Discovery** - Path discovery occurs for the first Interest received by the OMP-IF strategy of a CCN client at the beginning of a content download or after the expiration of all entries in the GFL for the given content.
- **Congestion and Packet Losses** - Congestion or the loss of packets (i.e., Interests or Content Objects) in the network may result in the CCN application adjusting the pace of the Interest flow through the operation of F&CC and possibly resulting in resending Interests.
- **Out-of-order Content Objects** - Due to delay differences in the paths being used, the Content Objects may not arrive in the same order in which the Interests were sent.
- **Caching** - The distance that an Interest has to travel depends on the availability of the requested content in the CCN routers along the path. The hop counts indicate how far the Interest had to travel.

The above listed influences affect the RTT and the hop counts. These in turn affect the operations of the path usage mechanism (i.e., splitting of Interest flow into multiple paths) at the CCN layer and the F&CC at the application.

Figures 5.14 and 5.15 show the progress of a single download for a content belonging to a popular class (i.e., Class 1) in terms of the Content Objects received for each of the Interests sent with Best-face and OMP-IF strategies, respectively. The progress is shown using 4 sub figures. The first 3 sub figures show the path usage (i.e., Faces used), RTT and hop count against the Content Object segment numbers. The last sub figure shows the arrival times of each Content Object (i.e., segment number against the normalized arrival time).

Both forwarding strategies send the first Interest (i.e., Segment 0) to all Faces to initiate the discovery of paths. Since RTT is the basis of selecting paths (i.e., Faces) to use, Best-face selects the Face 1 (Figures 5.14) while OMP-IF selects Face 1 and

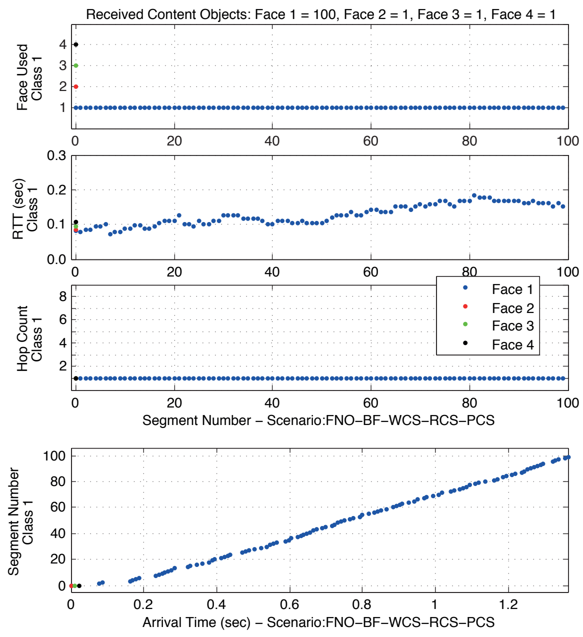


Figure 5.14: Progress of a Single Download of a Popular Class with Best-face Strategy in the FNO Scenario

Face 3 (Figure 5.15). Faces used sub figure shows the use of the selected Faces for the subsequent Interests.

Over the duration of the download, the RTT varies depending on the network conditions. Due to the use of the same path to send the Interests, the Best-face strategy congests the path thereby increasing the RTT. On the other hand, OMP-IF requests the Content Objects over multiple paths, balancing the ratio of the distribution of Interests considering RTT. Due to the real-time computation of the weights for the ratios, the OMP-IF prevents the buildup of congestion on the multiple paths which in turn prevents the gradual increase of the RTT. The final distribution ratio shows that 48 Interests were forwarded over Face 1 while 53 Interests were forwarded over Face 3.

A content of class 1 belongs to the most popular class. Therefore, it is likely that the Content Objects of a content of class 1 is available in most caches in the network. This is evident from the hop counts because all Content Objects in both forwarding strategies were retrieved from the CCN router found on the first hop

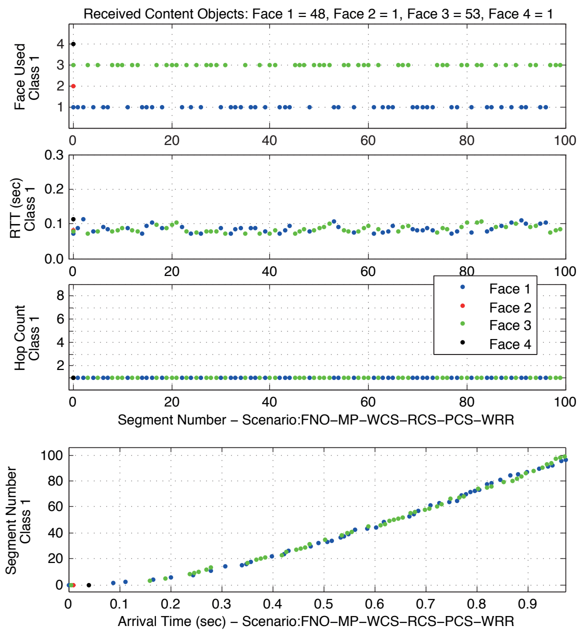


Figure 5.15: Progress of a Single Download from a Popular Class with OMP-IF Strategy in the FNO Scenario

(i.e., hop count is 1 for all Content Objects).

The arrival times of each Content Object show that Content Objects arrive at an increased pace with OMP-IF compared to Best-face. Therefore, the download time is smaller with OMP-IF.

Another observation is the differences seen with the receipts of out-of-order Content Objects. While Best-face shows no out-of-order receipts, OMP-IF shows out-of-order receipts spread across the download. This is due to the use of multi-path and the differing path characteristics. A closer investigation of the out-of-order segment numbers show that the out-of-order receipts are resolved within a maximum of 3 Content Object receipts. Since OMP-IF performs the distribution of Interests based on path characteristics (i.e., RTT) and as the OMP-IF strategy adjusts the distribution ratio as soon as the next Content Object arrives, the newly computed distribution ratio will guarantee that the path with out-of-order Content Objects is used less than the other path.

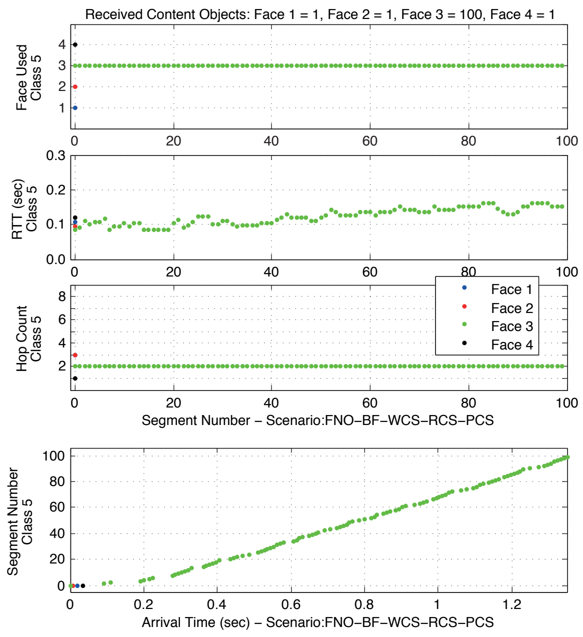


Figure 5.16: Progress of a Single Download from an Unpopular Class with Best-face Strategy in the FNO Scenario

Figures 5.16 and 5.17 show the download breakup results of a content for an unpopular class (i.e., Class 5). The results show that the Best-face strategy selecting a path with the content available 2 hops away as that path had the smallest RTT (Figure 5.16). The OMP-IF selects the 2 paths with the smallest RTT values which have the content 1 hop and 2 hops away (Figure 5.17).

The OMP-IF shows that the subsequent Content Objects arrive not only from the original cache found, but also from caches further away, as evident from the varying hop counts (i.e., between 1 and 4). This is a result of cache pollution and cache fragmentation (described in Section 3.3). These effects result in the RTT increasing. Since OMP-IF makes adjustments to the distribution ratio based on the RTT, the influence of these effects are minimized as seen from the subsequent RTT and the final download time (i.e. smaller download time). In the case of Best-face, these effects are not present as it most often downloads all the Content Objects from the same cache. But, similar to the observation with downloading contents of popular classes (shown in Figure 5.14), the effect of continuous increase of the

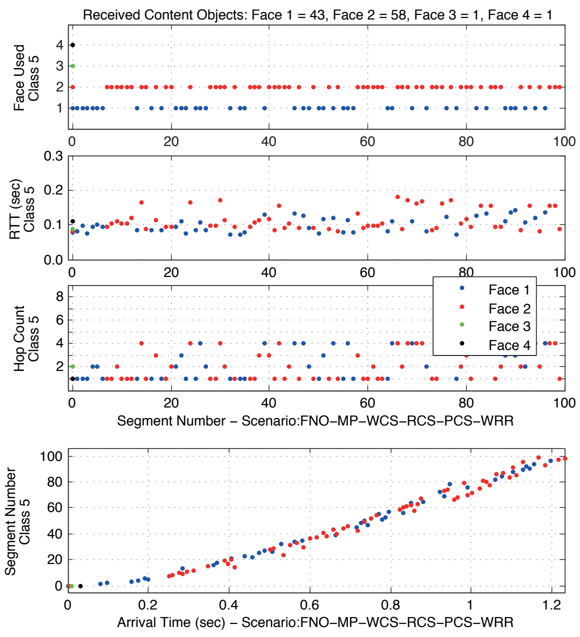


Figure 5.17: Progress of a Single Download from an Unpopular Class with OMP-IF Strategy in the FNO Scenario

RTT occurs for unpopular content as well.

5.5.4.2 Download Analysis - Zipf Classes

The CCN applications download contents based on the Zipf popularity model. Figure 5.18 shows the per class breakup of download time, RTT and throughput for Zipf classes 1 through 5. The fourth sub figure in Figure 5.18 shows the percentages of the class breakup of the overall download time.

Majority of the content downloads are for contents of class 1 (i.e., 60%). Contents of class 1 are downloaded the fastest as they are most often available at the first hop CCN router. Due to bandwidth aggregation and the use of WRR, the OMP-IF strategy results in the clients being able to download contents faster than the Best-face strategy. This in turn results in the clients with OMP-IF strategy being able to download more contents than Best-face strategy (e.g., 58,768 contents to 45,473 contents in class 1).

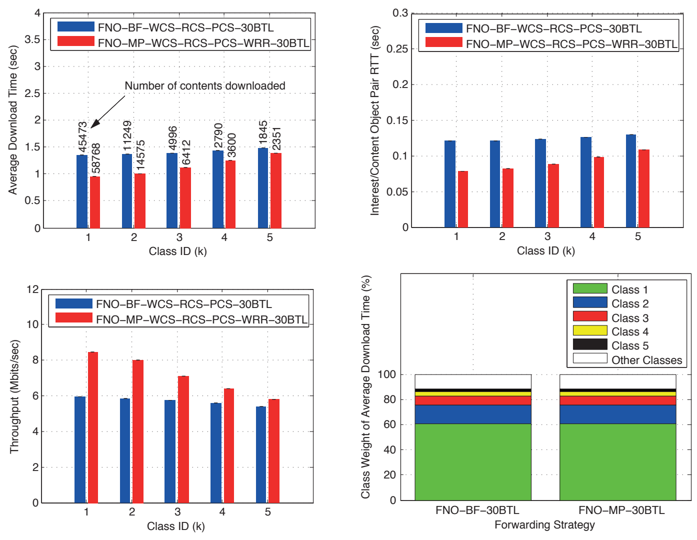


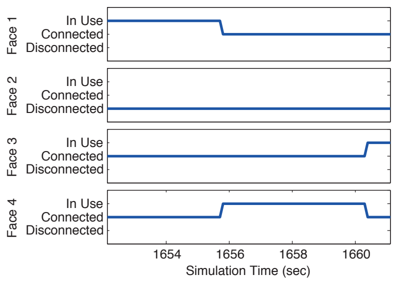
Figure 5.18: Per Class Performance for 30% BTL in the FNO Scenario

When the popularity of the class reduces (e.g., class 5), download times tend to increase as the Interests are required to travel further in most cases. This is due to the required Content Objects not being found in nearby caches. This results in the increase of the download times (as seen from the RTT). Though both strategies have increased download times, the gap between the two strategies tend to narrow as the popularity decreases. This is due to the effect of the cache pollution and cache fragmentation as explained in Section 5.5.4.1. Due to only parts of a content being available in the cache found at discovery, OMP-IF is affected by variances of the distances an Interest has to travel to reach the Content Object.

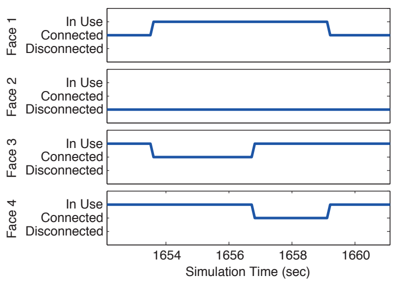
In summary, when considering the overall performance, OMP-IF performs the best (Figure 5.11(a)) as the Zipf based retrievals result in the majority of content downloads being for popular classes of content.

5.5.5 Performance in Mobile Network Operator (MNO) based Networks

The focus of this section is the performance analysis of the OMP-IF strategy and the best-face strategy when deployed in CCN client nodes operating in MNO based networks. Each of the CCN client nodes in the scenario uses the mobility model as described in Section 4.4.7.



(a) Best-face Strategy



(b) OMP-IF Strategy

Figure 5.19: Face Connectivity Status in MNO Scenario of a Selected CCN Client Node

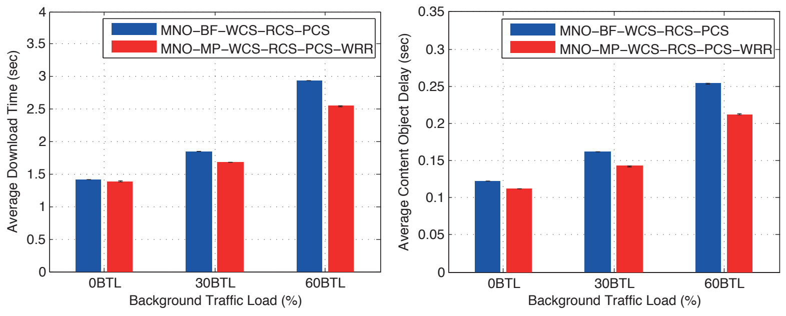
Figures 5.19(a) and 5.19(b) show the connectivity and usage status of the Faces of a CCN client node during mobility, equipped with 4 Faces. They show the status per forwarding strategy employed during a period of stable connectivity. As the mobility model uses the same Random Number Generator (RNG) seed in both forwarding strategies to make movement decisions, the connectivity of Faces have the same behavior. But, the usage behavior of the Faces differs due to the following reasons.

- Due to forwarding strategy employed: The best-face strategy uses only one Face at a time while OMP-IF uses up to 2 Faces, when possible. Figure 5.19(a) shows that only one Face is used to download a content at any given time. When a content is completely downloaded, the path discovery for the next requested content uses a different Face to download. Figure 5.19(b) shows a similar behavior, but uses 2 Faces simultaneously to download a content.

- Due to network conditions: Network conditions influence the speed at which a content is downloaded and which cache is used to retrieve the content. The latter point is due to the 2 forwarding strategies employing the RTT as the basis for selecting paths to use.

- Due to the type of content being downloaded: The content selected to download next is based on the Zipf popularity model. Therefore, if the identified content to download is a popular content (i.e., from a popular class), it is very likely that the content can be found in a cache closer to the CCN client node. Fetching an unpopular content may require the Interests to travel further, possibly ending up at the CCN server that hosts the content.

To simplify the visualization of *In Use* status of Faces, the Figures 5.19(a) and 5.19(b) do not take the path discovery phase into consideration. When the path discovery phase is considered, all *Connected* Faces in the selected FIB entry become *In Use* until the completion of the path discovery phase.



(a) Average Download Time

(b) Average Content Object Delay

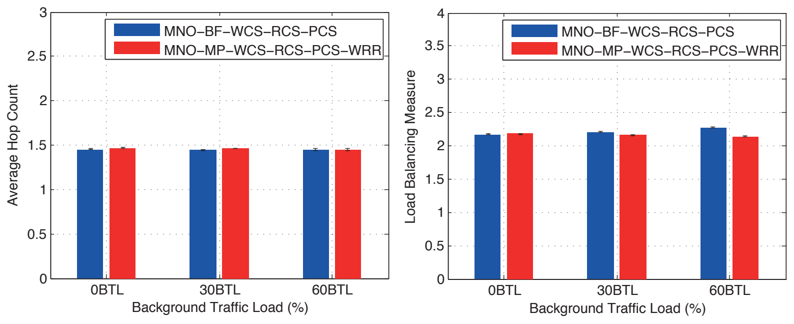
Figure 5.20: Performance of Download Time and Content Object Delay in MNO Scenario

Figure 5.20(a) shows the performance of Content Download Time under differing BTL levels in the mobility scenario. As in the FNO scenario, the download times increase with both strategies when the BTL is increased. But, the differences between the performance at different BTL levels are narrower than in the FNO scenario and further, the download times are higher. The reasons for higher download times and the narrower differences are as follows.

- Mobility of CCN client nodes results in the disconnection of Faces which were selected before the download of a content. There are 2 possible implications to the download based on the forwarding strategy employed. In the case of best-face strategy, another path must be discovered to send the Interests. This results in delays that come about due to the discovery phase as well as the expiration of currently pending Interests. Therefore, F&CC adjusts the *cwnd* to address the larger RTT values, thereby increasing the download times. In the case of OMP-IF strategy, when one of the currently active Faces is disconnected, the OMP-IF strategy continues to use the second Face that was part of the download. This results in the F&CC being effected due to the loss of the bandwidth aggregation that existed before the Face disconnection. Therefore, F&CC adjusts the *cwnd* to address the larger RTT values, thereby increasing the download times.

- Disconnection of a Face in a CCN client node using the OMP-IF strategy results in the strategy continuing to use the other Face that was active. This results in the OMP-IF strategy operating in the same manner as the best-face strategy, using a single path. As a result, the OMP-IF also shows similar download times as the best-face strategy. Another aspect that brings the best-face download times closer to the OMP-IF download times is due to the multiple path discoveries done by best-face during an active content download. When path discoveries are done in this manner, better paths can be found to retrieve a content.

There are 3 delay components influencing the download of a content. They are; (1) the delay associated with the time it takes for an Interests to travel to the CCN cache, (2) the delay associated with the Content Object that arrives at the CCN client node as a result of the sent Interest and (3) the effects of F&CC which considers perceived congestion in the network. Figure 5.20(b) shows how the second component, Content Object Delay is effected by mobility. Compared to the FNO scenario, the delays have become larger and the differences have narrowed. The delays shown in Figure 5.20(b) confirms the conclusions made previously with the Average Download Time of the MNO scenario.



(a) Average Hop Count (b) Load Balancing Measure

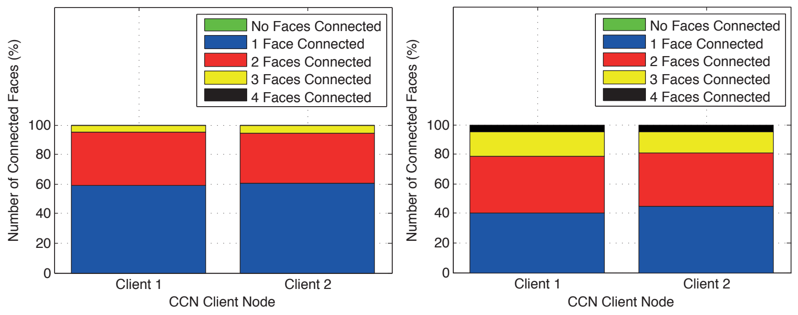
Figure 5.21: Performance of Hop Count and Load Balancing in MNO Scenario

Figure 5.21(a) shows the Average Hop Count in the MNO scenario. Compared to the FNO scenario, the hop counts at each BTL level has slightly increased.

Further, the margin of difference has also narrowed, but the best-face still has a slightly better performance than OMP-IF. The reason for the higher hop count with OMP-IF, compared to best-face, is due to the aggregation of hop counts from 2 paths which are node disjoint to each other. Due to the second path being a slightly longer path, the hop counts with the OMP-IF are higher than that of best-face. The reason for the slightly larger hop counts compared to the FNO scenario is due to the disruptions that occur with mobility. Without the disconnections experienced by the Faces, both forwarding strategies find the shortest paths. But, with mobility and hence, disconnections of Faces, the paths that are found may not be the optimum.

Figure 5.21(b) shows the comparison of the Load Balancing Measure with mobility of CCN client nodes for varying BTL levels. Compared to the FNO scenario, all values have degraded in general. The reason for this behavior is due to disruptions brought about by mobility. Due to mobility, the Faces are disconnected and due to these disconnections the paths that are made are not able to distribute the traffic more evenly throughout the network. Therefore, some parts of the network are never used.

As in the FNO scenario, OMP-IF in the MNO scenario performs better than best-face because of the use of multiple paths. The use of multiple paths to download content, together with the RTT based path usage mechanism helps to spread the traffic more evenly throughout the network. But, the margin of difference is comparatively smaller due to the OMP-IF being compelled to use a single path to download content due to the sparse availability of connectivity coverage of Faces (see Figure 5.5 for the coverage map).



(a) Number of Connected Faces (b) Number of Connected Faces (Extended Coverage)

Figure 5.22: Performance of Face Connectivity in the MNO Scenario

Figure 5.22(a) shows the percentage of times that different number of Faces had connectivity during the simulation time of the MNO scenario for 2 CCN client nodes. As explained before, it shows that a majority of the times (i.e., ~60%), both of these CCN client nodes were able to download contents only using one Face. Due to the availability of continuous WAN coverage, there is at least one connected Face for downloads. But, due to the way the coverage map is constructed in the MNO scenario (see Figure 5.5), there are no intersecting coverage areas that result in the availability of 4 simultaneously connected Faces. Therefore, almost 95% of the times, the CCN client nodes had between one and two Faces to use for downloading contents.

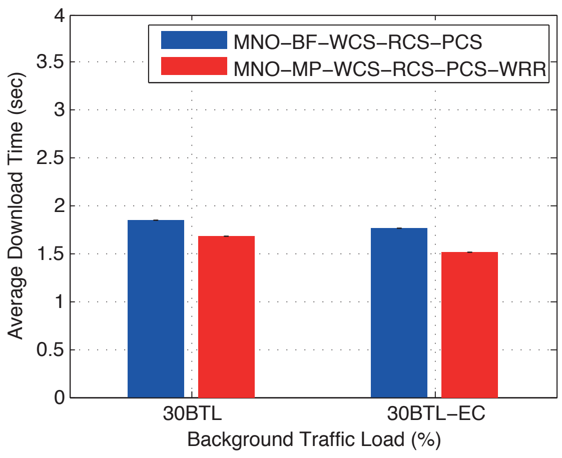


Figure 5.23: Performance of Download Time in the MNO Scenario with Extended Connectivity Coverage (EC refers to Extended Coverage)

Figure 5.22(b) shows the performance of the same 2 CCN client nodes when the coverage area is increased in one of the LANs. Thereby, the percentage of the availability of at least 2 Faces for the OMP-IF strategy to build multiple paths have increased. This in turn results in improved performance at the CCN client nodes as well as in the network. Figure 5.23 shows a performance comparison of the download times with and without the increased coverage area for both forwarding strategies under a 30% BTL level. Though both forwarding strategies show improved download times, the OMP-IF strategy shows a higher gain compared to the best-face strategy as the OMP-IF strategy is given more opportunities to create multiple paths to download content (i.e., OMP-IF has a gain of 10% while the gain with best-face is about 5%). Additionally, these improvements are also reflected in the load balancing measure.

Considering the analysis given above, it could be said that, unlike the performance improvements shown in stationary networks (i.e., FNO scenario), the performance of OMP-IF in mobile networks is highly dependent on the connectivity coverage of the Faces. Sparse coverage would have implications, not only on application performance but also on the performance of the overall network (i.e., in terms of load balancing).

5.5.6 Changes in Content Popularity

Most of the simulations performed in this thesis assume a steady state with CCN caches. To achieve the steady state in caches, the caches are pre-filled with Chunks considering the Zipf model values used, the cache sizes and Chunk sizes. In reality though, the popularity associated with contents may change over time. Such changes may result in certain contents becoming less popular and/or less popular (also new) contents becoming more popular.

To emulate the changes in status of the popularity of contents, i.e., Popularity Renewal (PR), a content belonging to a selected Zipf class is removed from all the caches in the network during the simulations. The following content removal cases are considered.

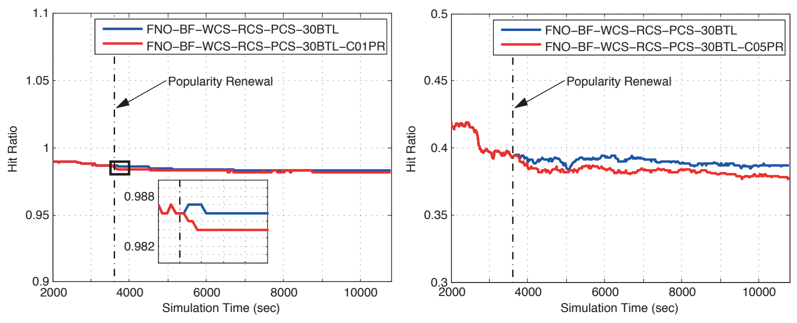
- Case 1: Removal of an extremely popular content, i.e., removal of all Chunks of one content belonging to Zipf class 1 which has a popularity of 60.8%.

- Case 2: Removal of a less popular content, i.e., removal of all Chunks of one content belonging to Zipf class 5 which has a popularity of 2.4%.

PR results in the caches requiring to rebuild themselves to represent the popularity model again. Once all the Chunks of the specified content are removed, the Interests for the dropped Chunks must reach the CCN server node to obtain the corresponding Content Objects. Thereby, gradually, each cache is made to rebuild the content representation of itself to the popularity model based on the Content Objects that travel through them.

Figures 5.24(a) and 5.24(b) show the performance of Hit Ratio with the above mentioned cases. They show that the Hit Ratio is effected when the PR occurs but the level and the duration of the effect differs based on the Zipf class.

As explained above, due to the renewals, Chunks are removed from the caches. This results in any request for these removed Chunks having to reach the CCN



(a) Zipf Class 1 (b) Zipf Class 5

Figure 5.24: Performance of the Hit Ratio with PR of two Contents, each belonging to two different Zipf Classes

server node. This, in effect, decreases the Hit Ratio in the caches because of the Chunk unavailability.

Figure 5.24(a) shows that the removal of the Chunks of a content of Zipf class 1 (identified by the tag **C01PR**) has very little effect on the Hit Ratio. This is due to the popularity level held by class 1. Class 1 has a 60.8% popularity and therefore, content requests are also very frequent as seen from the higher Hit Ratio (i.e., close to 1.0). Due to the higher request frequency, even though the Chunks are removed, the caches get re-populated faster, thereby showing a faster convergence of the Hit Ratio.

On the other hand, for a less popular class such as class 5, the frequency of requests are lower and therefore, the persistence of the effect of the Chunk removals lasts longer as seen from Figure 5.24(b) (identified by the tag **C05PR**).

6 Analytical Characterization of Multi-path Content Retrievals

The focal point of the previous chapters was on developing multi-path mechanisms for Content Centric Networking (CCN) and the evaluation of how these strategies performed in large scale scenarios. When considering the use of multi-path in CCN, another challenging area to consider is the analytical modeling of multi-path in CCN. There has been research done in the area of analytical characterization of data transfers in CCN ([CGLD11]). They focus on how caching affects the performance of content transfers in single path usage scenarios. [CGLD11] mainly focuses on extending the work done by [JK08] on analytical modeling of the Least Recently Used (LRU) cache replacement policy, to characterize CCN data transfers.

This chapter focusses on developing a multi-path analytical model extending the work done by [CGLD11]. The sections in this chapter are organized in the following manner. The related work section describes the model developed by [CGLD11] placing an emphasis on the aspects required by the extensions done in this work. The multi-path extensions section presents the extensions made to the [CGLD11] model to support multi-path content retrievals. The simulator setup section is a brief recap of the OPNET (Optimized Network Engineering Tools) environment on which the simulator model is built in this thesis (described in Chapter 4), with the emphasis on the aspects relevant to the evaluation of the analytical model. The last section provides the interpretation of the results obtained from the analytical model and the simulator for the single path ([CGLD11]) model and the multi-path model. The work done on the analytical model has been described in [UPG13].

6.1 Related Work

As indicated in the previous chapters, there is research being done in the area of multi-path performance in CCN networks. This work mainly focuses on simulated environments (e.g., [RR11], [YAW^{+}12] and [YAM^{+}13]) to evaluate performance. Analytical modeling of CCN can be considered as a challenging area due to the novelty of CCN, lack of research and the involvement of caching. When considering research done in analytical modeling for CCN, one of the primary publications in this area is the work done by the authors of [CGLD11]. In this research, they identify a model to analytically characterize data transfers in CCN. This work describes the model using scenarios in which single path data transfers are characterized.

Caching plays a major role in the performance of CCN since the load on the content providers' servers is reduced by the caches scattered all over the network [RR11]. The literature that describes caching converges primarily on LRU, Least Frequently Used (LFU) and Uniform algorithms for replacement policies. LRU is the most frequently used out of those as shown in [PCL$^+$11], [CGLD11] and [MCG11]. LRU is effective if one considers the fact that caches must store major portions of popular content, reduce server load and bandwidth usage [PCL$^+$11]. In comparison, the Uniform algorithm is good for its simplicity since LRU involves a processing time which increases proportional to the cache size. LFU on the other hand requires more processing than LRU as even the usage frequencies have to be maintained and compared.

The authors of [JK08] introduced a miss probability for the LRU cache with data having a Zipf popularity distribution [Wik35] and a Poisson request process. The authors of [CGLD11] have extended this single cache miss probability for a request process of a Markov Modulated Rate Process (MMRP), when contents are divided into classes with the Zipf popularity distribution. The basic concepts detailed in [CGLD11] are introduced in this section, in order to better understand further extensions done in this work to model multi-path behavior of CCN.

In the following explanations, the files that are transferred in the networks are identified as content and each content is subdivided into chunks. In CCN, a request for a single content is done by requesting all the chunks of that content, one after the other. The message that carries the Chunk is referred to as a Content Object and the payload of this message carries the Chunk. Further, the Zipf popularity distribution and the LRU cache replacement policy referred to in this thesis are defined in the following manner.

- **Zipf**: Contents are grouped into classes and each class is said to have a negative exponential relationship in terms of popularity with the adjoining classes. The form of this relationship between these classes is determined by the variable α. With an α of 1.0, the content of the first class is said to be two times more popular than the content of the second class, three times more popular than the content of the third class and so on. This relationship has been seen in empirical observations and subsequently formulated as a mathematical equation [Wik35]. Formulation of the Zipf popularity model is detailed in Section 4.3.3.

- **Least Recently Used (LRU)**: A caching policy where a least recently accessed Chunk in a cache is removed to make way for a Chunk that is required to be cached. The least recently accessed Chunks are determined by maintaining the last access times or by ordering them according to the previous accesses [Wik14] (e.g., using pointers).

6.1.1 Content Popularity and Topology

The contents in the model are divided into classes according to their popularities. The popularities follow the Zipf distribution [Wik35] with a request probability of $q_k = \frac{c}{k^\alpha}$ for the class k, where c is the normalization factor and $\alpha > 1$ determines the tail of the distribution. In other words, the files of class 1 have the highest probability of being requested and the request probability reduces with the increase of the class number.

To visualize the explanations of the single path model described in [CGLD11], a network topology with three clients, a cache and a server is used as shown in figure 6.1. The model assumes that all paths have the same characteristics and the clients request for content which are served by the caches or the content server.

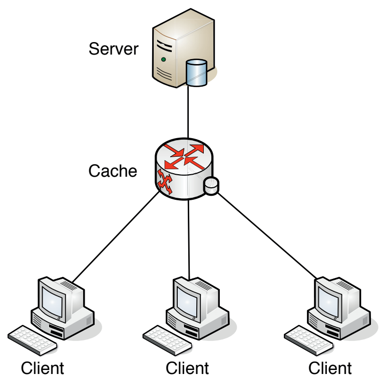


Figure 6.1: Single Cache Scenario

6.1.2 Content Size

The number of Chunks of a content is considered in [CGLD11] to be geometrically distributed with a probability of p_l representing the probability of an event where no Chunk is received (equation 6.1).

$$P(N_{ch} = a) = (1 - p_l)^a p_l \ \ \forall\, a \geq 1 \tag{6.1}$$

where N_{ch} is the total number of Chunk requests generated until the point when no Chunk is received. Based on this characterization, the size of a content is N_{ch}. Therefore, the mean size of a content, denoted by ω is expressed as in equation 6.2.

$$\omega = (\frac{1}{p_l} - 1) \tag{6.2}$$

Equations 6.1 and 6.2 also imply that every content should have at least one Chunk. Constant content sizes are not considered here, considering that in a real network scenario the content sizes are almost always random.

6.1.3 Arrival Process

The content request arrival process at a cache is considered as possessing Markov Modulated Rate Process (MMRP) [SE91] characteristics. The operation of an MMRP consists of two sub-processes. The first sub-process, which is a birth-death process has been equated to how requests for complete contents are generated by clients (i.e., content level). The second sub-process, which has deterministic characteristics has been equated to the request of individual Chunks of the content being requested (i.e., Chunk level). The general idea of using MMRP to characterize such a process is the notion that once a content download is commenced, all Chunks associated with the content are requested one after the other until the end.

At content level, the arrival process is a Poisson process with an arrival rate of λ_k for class k, where λ is the overall arrival rate and $\lambda_k = \lambda \times q_k$. At the Chunk level for a certain class, it is a deterministic process with Chunk requests for the same content being received at constant time intervals. These constant time intervals (referred to as the *Virtual Round Trip Time* (VRTT)) differ for each content class. In reality, the VRTTs are never constant as the basis on which the VRTTs are computed (i.e., hit/miss probabilities) may vary over time. Further, there could also be influences due to network conditions such as buffering in routers. In subsequent explanations however, a system that has reached the steady state is considered where the VRTTs converge to a constant value. Further details about MMRP with respect to the multi-path extensions proposed in this work are described in section 6.2.3.1.

6.1.4 Cache Miss Probability

Assuming the use of LRU (Section 6.1) to replace Chunks in caches, the MMRP based overall arrival process, geometrically distributed content sizes and the Zipf based popularity distribution, the authors of [CGLD11] have derived the following equation for the cache miss probability at Chunk level for a single path ($p_{k,sp}$) scenario.

$$p_{k,sp} \sim e^{-\frac{\lambda}{m} q_k g x^{\alpha}} \; \forall x > 0 \tag{6.3}$$

where $1/g = \lambda c \omega^{\alpha} m^{\alpha-1} \Gamma\left(1 - \frac{1}{\alpha}\right)^{\alpha}$, x is the cache size in Chunks and m is the number of contents in a class. g is assumed to be the same irrespective of single path or multi-path. This assumption, i.e., $g_{sp} = g_{mp} = g$ is proved in the equation 6.10 due to the simplifications done.

The intuitive idea behind formula 6.3 is that a request for a Chunk that arrives at a cache will produce a miss if more than x requests for unique Chunks have been received since the last request for the same Chunk. Therefore, $p_{k,sp}$ of a cache for the class k is influenced by the factors of class size, arrival rate, popularity of the class and the size of the cache. Figure 6.2 shows the miss probabilities of each class with different α values using equation 6.3 .

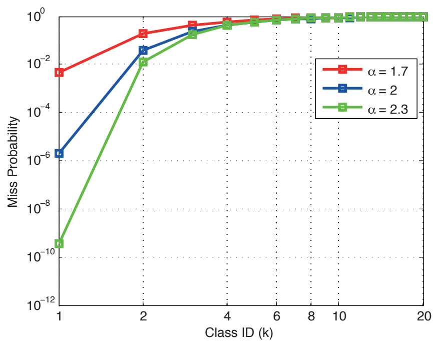


Figure 6.2: Cache miss probabilities for different α values

The value of α characterizes the form of the popularity curve in the Zipf based popularity model used in computing the miss probabilities in figure 6.2. Larger

α values characterize a content catalog where a few contents are extremely popular and therefore the miss probabilities are lower with these extremely popular contents (e.g., the curve with $\alpha = 2.3$). Smaller α values characterize content catalogs that have less extremely popular contents with the tail of the popularity curve showing better miss probabilities compared to the catalogs with higher α values (e.g., the curve with $\alpha = 1.7$).

6.2 Analytical Model

The analytical model that is proposed to characterize multi-path content retrievals consists of an extended arrival process and a cache miss probability that incorporates the existence of multiple paths from clients connecting to multiple caches. The assumptions and parameters explained in section 6.1 are applicable for this section and they are further elaborated with the inclusion of extensions for the multi-path model.

6.2.1 Topology

The general topology used for the multi-path model is given in figure 6.3. The caches to which the clients connect are called the first level caches. The cache level N is the server of the content producer. There are two basic assumptions made in this topology in terms of the connections between clients, caches and the server of the content producer. They are,

- Each client has an equal number of paths connecting to the caches and the number of paths are equal to the number of caches present at the level at which the clients connect (i.e., the first level in figure 6.3). This number of caches is denoted by Z. This implies that each client has Z paths to the caches. This further implies that the number of paths connected to a cache is equal to the number of clients present, which is denoted by Y.
- Each cache in this topology is considered to have only a single path to the next level cache or the server of the content provider.

6.2.2 Multi-path Usage Strategy

The multi-path usage strategy employed here is called *Splitting Strategy*. With this strategy, each Chunk of a content is requested and received over the number of available paths of a client in a round robin manner. This strategy can be used to

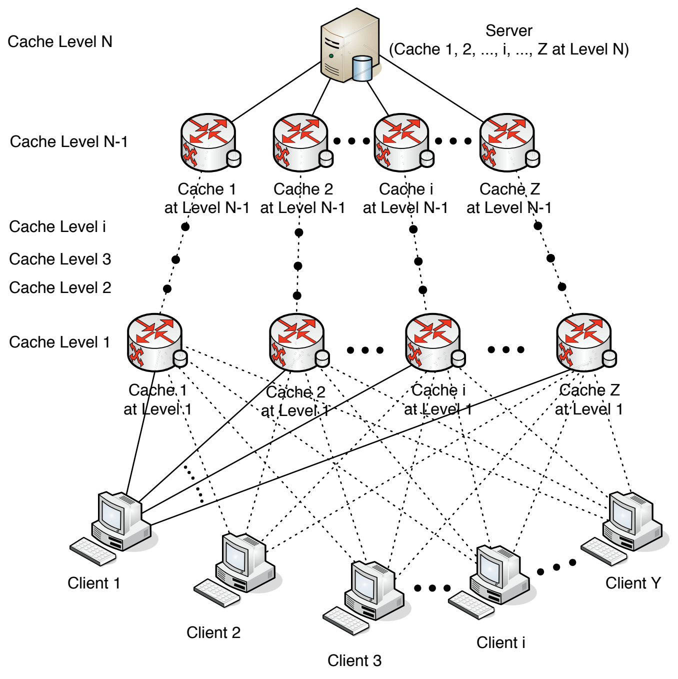


Figure 6.3: General Topology

improve the throughput and increase the overall efficiency based on other restrictions in the networks such as costs and bandwidth ([Zha11]).

Due to the operation of the *Splitting Strategy*, every cache that serves multiple clients is only requested for a part of a content by a client. Therefore, from the point of view of a cache, each of the caches only receives requests for a set of non-contiguous Chunks from a single client. Since these Chunks represent only a part of the content, these partial contents are identified as *Abstract Contents*. Further, since each Zipf class is considered to contain m contents, with the Z caches, each class now contains $m \cdot Z$ abstract contents.

6.2.3 Multi-Path Model

The multi-path content retrieval model proposed in this work, similar to the single path model explained in 6.1, characterizes the content class popularity using the Zipf distribution with the content sizes following a geometric distribution. The behavior of popularity is identical for each user and the caches employ the LRU mechanism as the cache replacement policy.

Based on the characterization described for the multi-path model in 6.2.1, the Arrival Process and the Cache Miss Probability has been extended to include the number of connected clients (Y) and the number of caches (Z) at each level to represent the behavior for multi-path content retrievals.

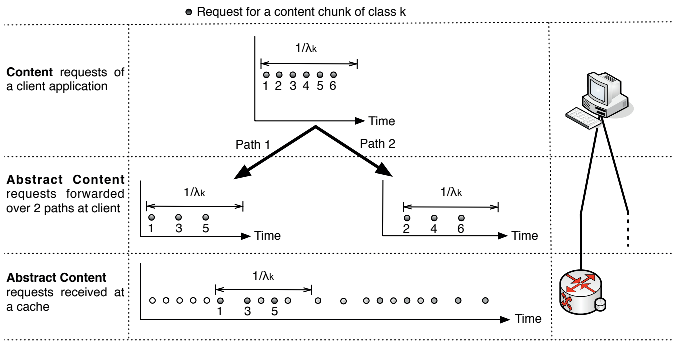

● Request for a content chunk of class k
Content requests of a client application
Abstract Content requests forwarded over 2 paths at client
Abstract Content requests received at a cache
1/λk
1 2 3 4 5 6
Time
Path 1
Path 2
1 3 5
2 4 6

Figure 6.4: Abstract Content Request Processes of a Client and at the Cache

6.2.3.1 Arrival Process

Similar to the content request arrival process in the single path model, a single request of an *Abstract Content* in the multi-path model is equally independent from the previous requests. Further, the client request process is sub-divided into a content request process consisting of *Abstract Contents* and a Chunk level request process. Therefore, the overall arrival process at a cache is MMRP with the arrival process at the content level for *Abstract Contents* being a Poisson process with λ_k and deterministic at the Chunk level with $\frac{1}{Z \cdot VRTT_k}$. Figure 6.4 shows a graphical view of the arrival process for a client with two paths and a cache that is connected to one of those paths. The application requests Chunks contiguously and these requests are distributed alternatively to the multiple paths. The cache connected to one of the paths receives the odd numbered Chunk requests, i.e., requests for the *Abstract Content*.

Since each client generates requests for *Abstract Contents* at a rate of λ_k for the k^{th} class, the aggregation of Y paths at a cache also produces a Poisson process. This occurs as a result of the superpositioning of Poisson processes, creating a Poisson process once more, with the arrival rates of each path being added. Hence the overall arrival rate at a cache becomes $Y\lambda$ and the arrival rate of the k^{th} class becomes $Y\lambda_k$. The Markov chain that models the number of ongoing *Abstract Content* downloads for a given class k is shown in figure 6.5.

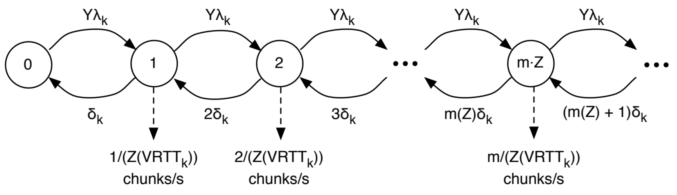


Figure 6.5: Markov Chain for the MMRP Process at a Cache

The states in the Markov chain represent the number of parallel abstract content requests at a given time with the assumption that the cache can process these requests simultaneously. The arrival rate does not depend on the state. At the state $m(Z)$, Chunks are downloaded at a rate of $\frac{m}{Z \cdot VRTT_k}$ Chunks per second. Simultaneous requests of Chunks from Z caches result in higher throughput and lower content download time for the client as described in section 6.2.4.

The rate of one *Abstract Content* being downloaded is denoted by the equation,

$$\delta_k = \frac{1}{\omega VRTT_k} \tag{6.4}$$

where δ_k is the service rate for class k. The service rate refers to the rate at which the Chunks are downloaded to complete the download of a content. The reasoning behind equation 6.4 is as follows.

With the availability of multiple caches (Z) to download from, a single Chunk is now downloaded with a duration of $Z(VRTT_k)$ due to the adoption of the splitting strategy. Similarly, since a client now splits the download between the available caches (Z), one path carries $\frac{\omega}{Z}$ Chunks. Therefore, the service rate (δ_k) is stated as in equation 6.5.

$$\delta_k = \frac{1}{(\frac{\omega}{Z})Z(VRTT_k)} \tag{6.5}$$

The simplification of equation 6.5 results in equation 6.4. The $VRTT_k$ of equation 6.4 is computed as in equation 6.6.

$$VRTT_k = \sum_{i=1}^{N} \left[R_i(1 - p_k(i)) \prod_{j=1}^{i-1} p_k(j) \right] \tag{6.6}$$

where R_i is the round trip delay between a client and the i^{th} cache and $p_k(i)$ is the miss probability of the i^{th} cache when there are N levels of caches (including the server).

6.2.3.2 Cache Miss Probability

The cache miss probability follows a similar characterization as in the single path model described by [CGLD11] but with extensions considering the influence of the Z caches and the Y clients.

As shown in figure 6.3, a single client has Z paths, each connected to one of the Z caches. Further, due to the deterministic Chunk request arrivals (section 6.2.3.1), all Chunk requests together are transmitted with a gap of $VRTT_k$ for the class k. Since all these requests are transmitted over all the paths of a single client, each path carries $\lceil \frac{Content\ size}{Chunk\ size \times Z} \rceil$ Chunk requests. The time gap between two requests on a path is $VRTT_k \cdot Z$ for class k.

The characterization of the content sizes and the typification of content in the multi-path environment is best explained with an example. Consider an example where a client is connected to two caches (i.e., $Z = 2$). Since the stream of Chunk requests are distributed to the multiple paths based on the *Splitting Strategy*, one of those paths would carry the odd numbered requests while the other would carry the even numbered requests. Therefore, from the perspective of a cache, the cache is serving an *Abstract Content* and not the whole content. As a consequence of the existence of *Abstract Contents* in each of the two caches, the number of abstract contents in a class becomes $2m$ and the average content size becomes $\frac{\omega}{2}$ Chunks. Even though the number of contents (i.e., *Abstract Contents*) have increased in a cache, the arrival process continues to hold the same characterization.

The miss probabilities of the single path equation 6.3 could be extended for the general multi-path case, where there are Z caches and Y clients. The number of *Abstract Content* in a class becomes $m \cdot Z$, the average *Abstract Content* size becomes $\frac{\omega}{Z}$ and the arrival rate at a cache becomes $Y\lambda$. The extended first level cache miss probability for a Chunk for multi-path ($p_{k,mp}(1)$) is now denoted by the equation,

$$p_{k,mp}(1) \sim e^{-\frac{Y\lambda}{Zm} q_k g x^{\alpha}} \ \forall x > 0 \tag{6.7}$$

where $1/g = Y\lambda c \left(\frac{\omega}{Z}\right)^{\alpha} (Zm)^{\alpha-1} \Gamma\left(1 - \frac{1}{\alpha}\right)^{\alpha}$. As indicated in Section 6.1.4, g is assumed to be the same irrespective of single path or multi-path. This assumption, i.e., $g_{sp} = g_{mp} = g$ is proved in the equation 6.10 due to the simplifications done.

The intuitive idea behind formula 6.7 is that a request for a Chunk that is received at a cache will produce a miss if more than x requests for unique Chunks have been received since the last request for the same Chunk. Unlike the single path formulation in equation 6.3, the differences for multi-path are the considerations made to accommodate arrivals from Y clients and the request for *Abstract Contents* (i.e., $Z \cdot m$).

When considering the operation of a cache in CCN, it is clear that a cache is unaware of the number of clients in the network or the paths of clients being served. Therefore, the miss probabilities of a cache can be considered as being independent of the Z caches and the Y clients in the network. This indicates that the outcome of equation 6.3 and equation 6.7 should result in the same miss probabilities. Therefore, equation 6.7 can be simplified as shown below to equation 6.3. For the simplification, equation 6.7 is rewritten as in 6.8.

$$p_{k,mp}(1) \sim e^{-A} \tag{6.8}$$

The value of A is expanded as in equation 6.9.

$$A = \frac{Y\lambda}{Zm} q_k x^{\alpha} \left[\frac{1}{Y\lambda c \left(\frac{\omega}{Z}\right)^{\alpha} (Zm)^{\alpha-1} \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}} \right] \tag{6.9}$$

The simplification of A is given below:

$$A = \frac{Y\lambda}{Zm} q_k x^{\alpha} \left[\frac{(Zm)\cdot(Zm)^{-\alpha}}{Y\lambda c \left(\frac{\omega}{Z}\right)^{\alpha} \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}} \right]$$

$$A = \frac{Y\lambda}{Zm} q_k x^{\alpha} \left[\frac{(Zm)\cdot(Z)^{-\alpha}\cdot(m)^{-\alpha}}{Y\lambda c (\omega)^{\alpha} \cdot Z^{-\alpha} \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}} \right]$$

$$A = q_k x^{\alpha} \left[\frac{(m)^{-\alpha}}{c(\omega)^{\alpha} \cdot \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}} \right]$$

$$A = q_k x^{\alpha} \left[\frac{1}{c(\omega m)^{\alpha} \cdot \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}} \right]$$

$$A = \frac{\lambda q_k x^{\alpha}}{m}[\frac{1}{\lambda c\,(\omega)^{\alpha}\,(m)^{\alpha-1}\,\Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}}]$$

$$A = \frac{\lambda}{m} q_k x^{\alpha}[\frac{1}{\lambda c\,(\omega)^{\alpha}\,(m)^{\alpha-1}\,\Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}}]$$

With the simplification of A, the first level cache miss probability for a Chunk for multi-path ($p_{k,mp}(1)$) is rewritten as in equation 6.10.

$$p_{k,mp}(1) \sim e^{-\frac{\lambda}{m} q_k g x^{\alpha}} \tag{6.10}$$

where $1/g = \lambda c \omega^{\alpha} m^{\alpha-1} \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}$. Therefore, when compared with equation 6.3, the following could be said about equation 6.10.

$$p_{k,mp}(1) = p_{k,sp}(1) \tag{6.11}$$

Due to the conclusion in equation 6.11, subsequent references to $p_{k,sp}$ and $p_{k,mp}$ are stated as p_k.

6.2.3.3 Second Level Cache Miss Probability

Once the first level cache miss probability for the class k, $p_k(1)$ is known, the i^{th} level cache miss probability for the class k, $p_k(i)$ in a topology with multi level caches is derived in [CGLD11]. $p_k(i)$ depends on the previous miss probabilities of the caches as shown in equation 6.12.

$$p_k(i) \equiv p_k(1)^{\Pi_{j=1}^{i-1} p_k(j)}\ \forall\, i > 1 \tag{6.12}$$

where i and j represent the cache level numbers. This is derived in [CGLD11] based on the proof of equation 6.13, which denotes the relationship between g and $g(i)$.

$$\frac{g}{g(i)} = \frac{\lambda(i)}{\mu(i-1)} \tag{6.13}$$

where $1/g = \lambda c \omega^{\alpha} m^{\alpha-1} \Gamma\left(1-\frac{1}{\alpha}\right)^{\alpha}$ as given earlier for the first level cache, $g(i)$ represents g of i^{th} level cache, $\lambda(i)$ represents the overall request arrival rate of the i^{th} cache and $\mu(i-1)$ represents the overall miss rate of the previous cache.

The following derivations show the applicability of equation 6.12 for the scenario as shown in figure 6.3, where all the caches in a branch are connected serially. Therefore, the arrival rate of the i^{th} cache should be equal to the miss rate of the previous cache. Under the assumption of an MMRP, the miss probability of the i^{th} cache depends on the request arrival process and the popularity distribution.

Considering that $i = 2$, based on the above description, the content request rate at the second level cache could be stated as follows.

$$\lambda(2) = \mu(1)$$

The value of $\mu(1)$ is computed based on the arrivals at the first level cache as follows.

$$\mu(1) = \lambda[p_1(1)q_1 + p_2(1)q_2 + ... + p_k(1)q_k]$$

where $\lambda = \lambda(1)$ is the overall request arrival rate of the first level cache. Based on the above characterizations, $\lambda(2)$ is written as follows.

$$\lambda(2) = \lambda \sum_{j=1}^{K} p_j(1)q_j$$

Popularity distribution of the class k at the second level cache, $q_k(2)$ is computed as follows.

$$q_k(2) = \frac{p_k(1)q_k}{p_1(1)q_1 + p_2(1)q_2 + \ ... \ + p_k(1)q_k}$$
$$q_k(2) = \frac{p_k(1)q_k}{\sum_{j=1}^{K} p_j(1)q_j}$$

By applying $\lambda(2)$ and $q_k(2)$ to 6.3, the miss probabilities of the second level cache is obtained in the following manner.

$$p_k(2) = e^{-\frac{\lambda(2)}{m} q_k(2) g(2) x^{\alpha}}$$

$$p_k(2) = e^{-\frac{\lambda}{m} \sum_{j=1}^{K} p_j(1) q_j \left(\frac{p_k(1) q_k}{\sum_{j=1}^{K} p_j(1) q_j} \right) g(2) x^{\alpha}}$$

$$p_k(2) = e^{-\frac{\lambda}{m} p_k(1) q_k g(2) x^{\alpha}}$$
$$p_k(2) = e^{-\frac{\lambda}{m} q_k g x^{\alpha} \left(\frac{g(2)}{g}\right) p_k(1)}$$

Since $e^{-\frac{\lambda}{m} q_k g x^{\alpha}}$ (equation 6.3) is the first cache miss probability, $p_k(2)$ is re-written as follows,

$$p_k(2) = p_k(1)^{\left(\frac{g(2)}{g}\right) p_k(1)}$$

Simplifying the above equation further, the equation 6.13 can be written as $\frac{g}{g(i)} = 1$ for the considered topology since $\lambda(i) = \mu(i-1)$. Based on this simplification, the miss probability of the second level cache can be re-written as in equation 6.14.

$$p_k(2) = p_k(1)^{p_k(1)} \tag{6.14}$$

6.2.4 Performance Metrics

The performance of the extended arrival process and the cache miss probability of the model is evaluated using a set of metrics. The following subsections identify these metrics and the simplifications made to enable the evaluation.

6.2.4.1 VRTT

The equation 6.6 is the general form of obtaining $VRTT$ considering a topology where there are N levels of caches on a certain path including the content provider. When considering a simpler topology with a single cache between the client and the server of the content provider, this expression can be simplified by regarding the round trip delays between the links (i.e., the two links between client, cache and server) as being equal. The $VRTT$ of class k for a scenario where there is only one level of caches between the server and a client is expressed in the following manner.

$$
\begin{aligned}
VRTT_k &= \sum_{i=1}^{2} R_i(1-p_k(i)) \prod_{j=1}^{i-1} p_k(j) \\
VRTT_k &= R_1\,(1-p_k(1)) + R_2\,(1-p_k(2))\,p_k(1) \\
VRTT_k &= R_1\,(1-p_k(1)) + 2R_1 p_k(1)
\end{aligned}
$$

where $p_k(2)$ = 0 and R_2 = $2R_1$. R_1 is the round trip delay between client and the cache and R_2 is the round trip delay between client and the server.

6.2.4.2 Throughput

Throughput (X_k) refers to the number of Chunks of content of the class k received by a client during a certain period of time and stated in Chunks/sec. Therefore, the average stationary throughput of class k for the single path case is computed as in equation 6.15, with the assumption of path delays being influenced only by link delays (i.e., no influence of network congestion, buffering delays, etc.).

$$X_{k,sp} = \frac{W}{VRTT_k} \tag{6.15}$$

where W is the window size maintained at the client, which determines the number of parallel requests issued.

Equation 6.15 can further be extended with the inclusion of the number of paths (Z) to download Chunks simultaneously in the multi-path case.

$$X_{k,mp} = \frac{Z \times W}{VRTT_k} \tag{6.16}$$

where W is the window size maintained at the client, which determines the number of parallel requests issued and Z is the number of parallel paths a client is able to use to download content.

6.2.4.3 Download Times

Download time is the average time taken for all the Chunks of a content in a certain class to reach the client. This is expressed in equation 6.17 for the single path ($T_{k,sp}$) and equation 6.18 for the multi-path ($T_{k,mp}$) cases.

$$T_{k,sp} = VRTT_k \times \omega \tag{6.17}$$

$$T_{k,mp} = \frac{VRTT_k \times \omega}{Z} \tag{6.18}$$

where $VRTT_k$ is the round trip time associated with the class k, ω is the average content size in Chunks and Z is the number of paths used to download Chunks simultaneously in the multi-path case.

6.3 Simulation Model

The validation of the multi-path analytical model is performed using the packet level OPNET based CCN simulator detailed in Chapter 4. This section provides a brief reminder to the protocol layers relevant for evaluating the analytical model including the parameters used.

6.3.1 CCN Application Layer

The CCN based applications operate at this layer. Each application generates and/or consumes CCN interests and/or content depending on the type of application. Each application operates either in the server mode or client mode. There are a number of different applications operating at this layer including applications that support flow and congestion control.

For the purpose of validating the analytical model, an application has been built to generate requests for content using the Zipf popularity model, where each content is grouped into classes. There are 2,000 Zipf classes and each class has 10 different contents. The names of content carry a unique number that is uniformly distributed between 1 and 10, inclusive. The prefix of a content carries a number representing the class of the content which is a number between 1 and 2,000, inclusive, and Zipf distributed with parameter α set to 2.0. Choosing the value for α is also critical in modeling the popularity since both the amplitude and the tail for different classes are affected based on the value selected. In [RR11], it is mentioned that the value for α can vary between 0.6 and 2.5. Lower value ranges of α model web servers with smaller content catalogs while higher value ranges represent web servers with large content catalogs such as Youtube. There is no clear agreement as to which value should be taken to assess the performance of CCN. In these evaluations a value of 2.0 is chosen due to the work done in [CKR$^+$07] which characterizes the behavior of content downloads at YouTube. The content sizes are geometrically distributed with a mean of 100 Chunks per content and each Chunk is 10,000 bytes long.

Parameter	Description	Value Used
$m_k = m$	Number of contents in a class	10
x	Cache size in Chunks	5,000
ω	Mean content size in Chunks	100
λ	Arrival rate in content/second	40
$q_k = \frac{c}{k^\alpha}$	Zipf probability	c = 0.608112 α = 2.0
K	Total number of classes	2000
R_1	Round trip delay at the link layer to the first cache in seconds	0.002
Y	Number of users	1
Z	Number of paths	2
N	Number of caches	1 and 2
W	Window size at the client	1

Table 6.1: Parameter Values Used in the Analytical and Simulated Models

6.3.2 CCN Layer

The CCN layer implements the core functionality of CCN to forward *Interests* and *Content*. This functionality includes the management of a number of CCN elements (Pending Interest Table, Content Store, etc.) and the operation of the forwarding strategy.

The CCN layer implements a number of forwarding strategies ranging from the standard forwarding strategy ([JST$^+$09]) to different multi-path forwarding strategies. For the purpose of evaluating the analytical model, a multi-path strategy that splits the requests for content to multiple paths is used (i.e., *Splitting Strategy*). The *Splitting Strategy* uses the Round Robin (RR) mechanism to distribute the Chunks of content between the multiple paths.

The CCN layer further implements a number of Content Store management policies that include different cache replacement policies such as LRU, First In First Out (FIFO), Uniform Random Replacement and cache decision policies such as Cache Always (ALWAYS), Random with a Fixed Probability (FIX(P)) [RR11]. The LRU implementation is used in the simulator to compare performance with the analytical model. The LRU cache replacement policy removes the least recently used Chunk of a cache when a new Chunk is required to be saved in the cache.

The parameter values used in analytical and simulated models are listed in the

Table 6.1. The Zipf model parameter values used are the same as the values used in evaluating the multi-path mechanisms proposed in this thesis with the large scale scenarios (Chapter 5). The differences relate to the use of small network topologies and applications without Flow and Congestion Control (F&CC). Each simulation is executed for 60,000 seconds. Due to the requests for content being generated based on the Zipf popularity model, the amount of requests received by unpopular classes are lower than the amount of requests for popular classes. Therefore, to avoid statistical effects (e.g., fluctuations) visible during short simulation runs, a larger simulation time (i.e., 60,000 seconds) is used.

A single simulation case (listed in Section 6.5) is executed 10 times with differing seed values to obtain the confidence intervals according to the *Student-t* distribution. The Appendix B lists the mean values and their 95% confidence interval values for the results shown in this chapter.

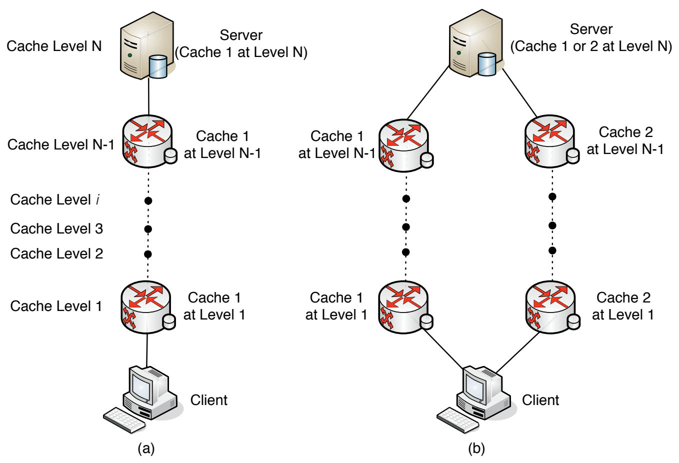


Figure 6.6: Simulated Single path (a) and Multi-path (b) Topologies

6.4 Validation of the MP Model

This section details the validation of equation 6.3 and equation 6.7 through simulations with respect to the hit ratio of a class derived from the served cache requests of a class to the total requests received. Figure 6.7 shows the hit ratios for simulated SP and MP against the mathematically computed values of all the classes.

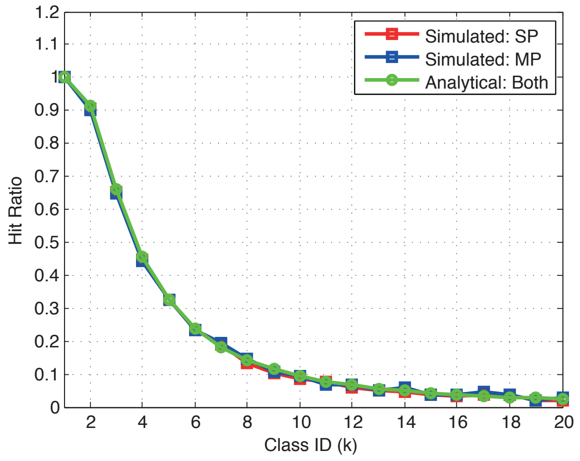


Figure 6.7: Hit Ratio Comparison for All Classes (Simulated and Analytical with Single Level of Caches)

Figure 6.7 shows that the hit ratios closely match between analytical and simulated, irrespective of SP or MP. In some classes, the results show a slight deviation in the simulated results. Further, the results show that this deviation is more evident in the higher numbered classes. This is due to the behaviour of Zipf and the duration of the simulations. The Zipf based content requests result in more requests arriving for popular classes (lower numbered classes) thereby closely matching the analytical results while the unpopular classes show deviations due to the receipt of a lower number of requests. The analytical values for the hit ratio are computed based on the equation 6.3 and equation 6.7. Figure 6.8 shows the progress of the hit ratio for a selected set of classes over time.

Figure 6.8 shows the hit ratio progressions for 4 classes that have differing popularity levels. These results are obtained after the elapse of 300 seconds of the simulation time. A 300 second margin is considered to let the content download application finish its initializations and to start the commencement of downloading. A further aspect is the way the hit ration results are collected. The results are collected periodically and therefore, the hit ratio is a cumulative of a number of hits or misses that occurred before the last collection time.

All curves from simulated results show fluctuations at the beginning due to the few content requests received for the corresponding class. Subsequently, the curves tend to smoothen and lie close to the analytical values. With the graph for

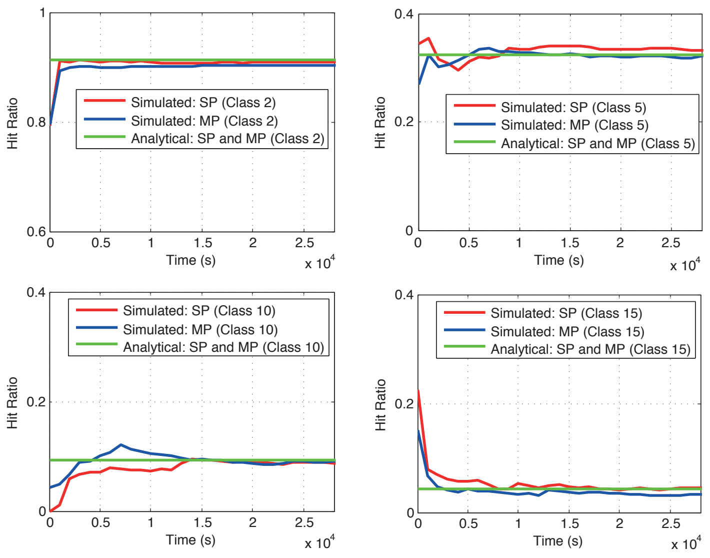


Figure 6.8: Hit Ratio Comparison for Selected Classes (Simulated and Analytical with Single Level of Caches)

class 15, it can be seen that the curves still tend to fluctuate compared to the others. This is due to the insufficient content requests received.

Another observation in these 4 graphs of figure 6.8 is the differing starting points of the curves with the simulated results. The reason for this observation is as follows. As stated above, the hit ratios are collected periodically and also after the elapse of 300 seconds. This way of collecting results together with the randomness of the requests that were generated for a particular class, the starting point of the hit ratio may either be above (e.g., class 15), below (e.g., class 10) or at the exact position relevant for the class.

6.5 Performance Results and Analysis

The performance results are obtained from the simulated and the analytical models. The results consider two scenarios based on the topologies shown in figure 6.6 for the following cases.

- Single path with a single level of caches

- Single path with two levels of caches
- Multi-path with a single level of caches
- Multi-path with two levels of caches

Each graph shows the average values for each class plotted for the first 20 classes. Table 6.2 lists the identifications assigned to the different scenarios in the graphs.

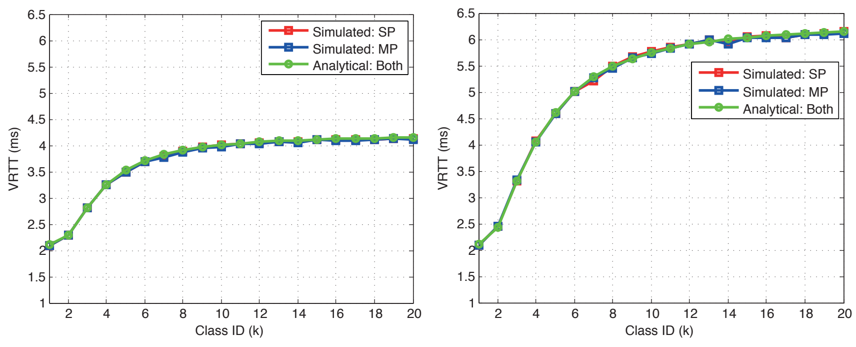
Scenario	**Analytical**	**Simulated**
Single path	Analytical:SP	Simulated:SP
Multi-path	Analytical:MP	Simulated:MP

Table 6.2: Scenarios Considered and the Legend Identifications

6.5.1 VRTT

Figure 6.9 shows the behavior of the $VRTT$ for each class with the cases of single level (a) and two levels of caches (b). The Zipf based popularity model shows the negative relationship between classes and the popularity of those classes, viz., the increase of the class order is related to a decrease of the popularity. This figure shows this nature where shrinking class popularities result in the increase of $VRTT$ values. This is due to caches being populated with only the popular content and thereby making the requests for less popular content travel further to fetch the content, possibly ending up at the content provider (server). When comparing the performance between the single level cache and two level cache scenarios, the values show that the $VRTT$ is close for popular classes and then tends to widen when the class numbers increase. The reason for this behavior is due to the nearest caches being able to serve popular content more frequently than unpopular content. In this way, $VRTT$s for popular content have more hits at the nearest cache irrespective of whether there is one or multiple caches before the server. But for unpopular content, the $VRTT$s increase as the Chunk requests have to travel longer in the two level cache scenario.

As the class popularity decreases the gap between the simulation and model results tends to widen slightly. This occurs as a result of the first few classes having a higher request ratio due to their very high popularity based on the Zipf distribution. As more and more content requests arrive for a given class the VRTT values tend to get closer to the value that the model represents. Observations made



(a) Single Level of Caches (b) Two Levels of Caches

Figure 6.9: VRTT Comparison (Simulated and Modelled)

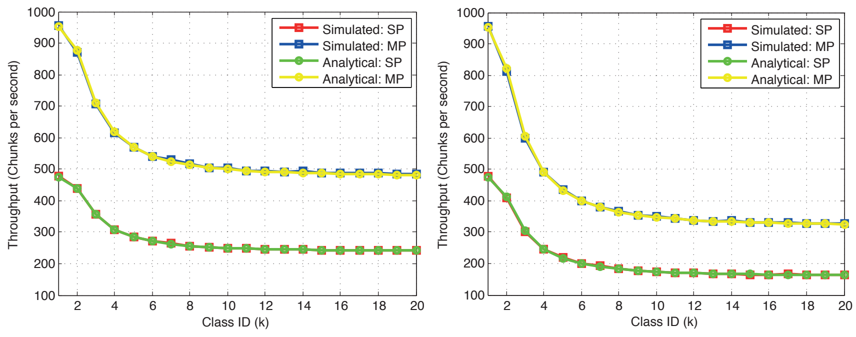
from simulations show that larger simulation times result in better alignment of the analytical and simulated results.

6.5.2 Throughput

The throughput in the simulation is computed based on the number of contents downloaded and the times taken to download these contents (explained in Section 6.2.4.2). These values are collected per class and then used to compute the average throughput for each of the classes. The analytical values for throughput is computed using the equation 6.15 for single path and the equation 6.16 for multipath.

Figure 6.10 shows the analytical and simulated results for a single level cache and two level cache cases for the different classes based on the scenarios listed in Table 6.2.

The results show that the throughput of the multi-path scenario (MP) is twice as high compared to the single path scenario (SP). This is due to the aggregation of the bandwidth through the use of the multiple paths, which results in the download times becoming twice as low (shown in figure 6.11). Another observation is the pattern that all curves follow. They have an inverse relationship to the Zipf distribution, indicating that downloads of popular classes have a better throughput performance due to the closeness of the content to the client. A further observation is the differences of the values in single level cache and two level cache cases, where the single level cache case shows a higher throughput. This is due to the



(a) Single Level of Caches (b) Two Levels of Caches

Figure 6.10: Comparison of Modelled and Simulated Throughput

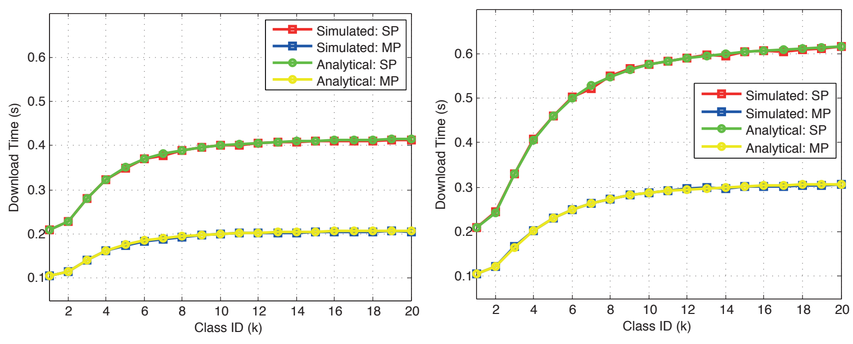
server being only two hops away, while in the the two level cache case, the server is three hops away. But, as the figure shows, this observation is not applicable to class 1 due to that class being the most popular class and hence, being served almost all the time by the first level cache.

6.5.3 Download Times

Figure 6.11 shows the behavior of download time (explained in 6.2.4.3) with the single level and two level cache cases for the scenarios mentioned in Table 6.2. Download times are plotted for both the model and simulation results for their corresponding class numbers. The download times in simulations are computed by collecting the individual content download times and computing the average times per class number. The analytical values for download times are computed using the equation 6.17 for single path and the equation 6.18 for multi-path.

The time taken for a download to finish in the multi-path (MP) scenario is twice as low, compared to the other scenario. This is due to the simultaneous use of multiple paths to request Chunks in the multi-path scenario, thereby resulting in twice as fast download times.

In all the results shown previously, one major observation is the close match that analytical and simulated models have for the different classes. As the class number increases, the simulated model results tend to fluctuate around the analytical model results. This is due to the effect of the higher numbered classes being less popular and thereby having a lesser number of content request arrivals to compute the mean.



(a) Single Level of Caches (b) Two Levels of Caches

Figure 6.11: Comparison of Modelled and Simulated Download Times

An additional observation visible in results of throughput and download times is the values for the first class. The results for MP and SP with respect to the first class are equal irrespective of whether it is the single cache case (Figure 6.11(a)) or two cache case (Figure 6.11(b)). This is due to the popularity of the Chunks in first class being extremely high which results in the first cache always being able to serve them, even in the two cache case (almost 100% hit ratio as seen from figure 6.7).

6.5.4 Overall Observations

The work in this chapter presented an analytical model to characterize multi-path content retrievals in CCN. The model extends the work done in [JK08] and [CGLD11]. A summary of the main observations from the work presented in this chapter are given below.

- The simulated hit ratios for single path and multi-path, and in single and two level cache scenarios show a close match with results obtained from the analytical model. This has been verified at the per class level and as well as at the overall network level (i.e., combined caches).
- The performance of the analytically modeled VRTT shows a close match with the simulated single path and multi-path results. The VRTT performance of the single level and two level caches differ due to the content server begin closer in the single cache level scenario compared to the two

level cache scenario. These differences occur mostly at the higher numbered classes (i.e., unpopular classes) as the content of popular classes are very likely to be present at the first level cache resulting in similar VRTT values (Figure 6.9).

- The analytically modeled throughput and the simulated throughput closely match as seen from Figure 6.10. The use of multi-path and bandwidth aggregation results in twice the bandwidth compared to single path. Content downloads of popular classes have a better performance (i.e., higher bandwidth) compared to unpopular classes in both single and multi-path scenarios. This is due to the contents of popular classes most often arriving from the first level cache.
- The download times in the analytical model closely matches with the simulated results. Due to the effect of the bandwidth aggregation, the multipath downloads are twice as fast as the single path downloads. Due to the likely arrival of the contents of popular classes from the first level caches, the popular classes have better download times compared to unpopular classes (Figure 6.11).

7 Summary and Conclusions

The work presented in this thesis focuses on incorporating multi-path mechanisms in Content Centric Networking (CCN) and evaluating the performance. The purpose of this chapter is to summarize the work done, provide conclusions and to discuss areas of improvements for the future.

7.1 Summary

The architecture of CCN, by design, has multi-path capabilities due to its support for loop-free forwarding and distributed caching. The creators of CCN propose 2 forwarding strategies in [JST$^+$09]. A CCN node deployed with the standard strategy broadcasts the Interests received to all the Faces associated with a selected Forwarding Information Base (FIB) entry. The best-face strategy uses a single Face from the selected FIB entry after determining the best Face to retrieve the content.

The standard strategy which replicates Interests results in the increase of load in networks while the best-face strategy experiences performance degradations due to the continuous use of a single Face. Therefore, these strategies are unsuitable for use as they drag the performance of networks down and in turn, degrades the performance in user applications. The purpose of this thesis is to identify and evaluate multi-path mechanisms to overcome these issues by efficient utilization of network resources.

Diversity is a common criterion considered in selecting multiple paths for better load balancing in networks and to improve the performance of applications through bandwidth aggregation. Since the selection and use of disjoint paths provide a higher aggregated bandwidth and fault tolerance, this thesis focuses on identifying a set of mechanisms to discover and use node disjoint multiple paths. Current research related to multi-path is mainly present in Internet Protocol (TCP/IP) based networks and the research related to disjoint paths use is mainly limited to research in TCP/IP based Ad-hoc networks which have dynamic network topologies. Research, such as [RR11] and [YAM$^+$13] discuss multi-path for CCN but do not focus on disjoint path discovery and usage.

Unlike in TCP/IP based networks, in CCN based networks, routing is independent from the identities or addresses of hosts. Conventional source routing protocols and link state protocols which rely on identities or addresses of hosts are unsuitable for CCN to discover node disjoint paths. But, the research done in [YKT03] and [MD01] related to the Ad hoc On-Demand Distance Vector (AODV) protocol provides a basis for developing on-demand, hop-by-hop path discovery mechanisms for CCN.

The work done in this thesis focuses on developing and evaluating multi-path mechanisms for CCN. Additionally, these mechanisms are formulated as a mathematical model and compared with the performance of the simulation model. The proposed multi-path mechanisms are used to discover, use and maintain node disjoint multiple paths to retrieve content in CCN. Following is a brief description of the work done.

- Identified a set of mechanisms related to path discovery, use and maintenance of multi-path in CCN called the On-demand Multi-path - Interest Forwarding (OMP-IF) strategy. It includes the node disjoint path discovery and the delay based path usage mechanisms to split the Interest flow.

- Identified an analytical model to characterize multi-path content retrievals in CCN. It is based on a Markov Modulated Rate Process (MMRP) model with the arrival process characterized through Poisson and Deterministic processes. The Content Store (CS) behavior is modeled using the Least Recently Used (LRU) caching model.

- Developed the complete CCN protocol stack in the Optimized Network Engineering Tools (OPNET) simulator, deployed with the functionality described in [JST$^+$09] for CCN client nodes, CCN server nodes and CCN router nodes. In addition to the mechanisms described in [JST$^+$09], it also includes a Zipf based traffic model, an LRU based real CS model, an LRU based mathematical CS model, a Link State Routing (LSR) based FIB population mechanism, a threshold based Pending Interest Table (PIT) expiration mechanism, a random direction based mobility model and Flow and Congestion Control (F&CC) based applications.

- The performance evaluations of OMP-IF include the comparison of performance with the best-face forwarding strategy proposed in [JST$^+$09].

- The evaluation scenarios consider large scale network operator based networks focusing on Fixed Network Operator (FNO) type networks and Mobile Network Operator (MNO) type networks.

- The topology used in the evaluations is based on the network topology described in [HPSS03] related to a Point of Presence (PoP) topology of AT&T.

7.2 Observations and Conclusions

The performance of the OMP-IF strategy is evaluated in the simulation model using the FNO and MNO scenarios which show differing performance results. The observations and conclusions are as follows.

- The OMP-IF strategy discovers the optimal multiple paths and balances the traffic between the selected paths based on the delays experienced, thereby maximizing the benefits of bandwidth aggregation. Table 7.1 shows the performance increase comparison for differing Background Traffic Loads (BTLs). The comparison shows that, the more the network is congested, the better the OMP-IF performs. The reasons for this behavior is twofold. Firstly, OMP-IF discovers node disjoint uncongested paths and secondly, it performs a real-time balancing of the distribution of the Interests between the paths to minimize the effects of delays.

- Though, the best-face strategy is able to identify an optimal path at the beginning of a content download, subsequent use of the same path results in increased congestion that in turn degrades the performance of applications. The OMP-IF strategy on the other hand, performs better than the best-face strategy as it distributes the traffic to the multiple paths by dynamically adjusting the ratio, reducing the chances of congestion building up along any selected path.

- Due to the node disjoint path discovery and the use of the paths based on delay, the OMP-IF strategy ensures that the distribution of the traffic throughout the network is always better compared to the best-face strategy.

- The performance in the MNO scenario shows that the OMP-IF strategy performs better than the best-face strategy albeit the overall performance has degraded compared to the FNO scenario. This is due to the effects of mobility that results in Face disconnections experienced by the CCN nodes during mobility. A further observation is the narrowed differences of performance of the 2 strategies. This is due to the amount of time that the OMP-IF is made to use one Face due to Face disconnections. Improvements in connectivity coverage improves the performance of OMP-IF.

- The use of modeled caching based on the analytical model is not suitable for F&CC based applications. The analytical model identifies a set of probabilities (i.e., per content class) to determine the availability of Chunks in a cache. Due to the use of randomness in determining the actual availability of a Chunk within the probabilities identified, there are instances when Chunks become abruptly unavailable after a stream of Chunk availabilities in the cache. This behavior results in F&CC reacting to reduce the flow of Interests, thereby reducing the overall throughput of the application.

- The Weighted Round Robin (WRR) mechanism of distributing Interests to multiple paths performs better than the Leftover Capacity Utilization (LCU) mechanism. The LCU mechanism, which sends Interests over the last Face that brought back a Content Object, has the possibility of clogging an already clogged path further, thereby adversely influencing the performance of the application and the network. The WRR mechanism, which maintains a ratio of distribution based on the Round Trip Times (RTTs) of the previously received Interest-Content Object pairs, adjusts the distribution ratio according to network conditions, thereby efficiently using the selected paths.

- Popularity Renewals (PRs) show that changes to popularity of contents have an effect on the downloads in terms of the cache hit ratio and thereby, the download times. After a PR, downloads for popular classes make caches converge faster than the downloads for less popular (or unpopular) classes due to the frequency of requests received.

Background Traffic Load	**Forwarding Strategy**		**Gain**
	Best-face	**OMP-IF**	
No BTL	1.064097 sec	0.900476 sec	15.4% ↑
30% BTL	1.403168 sec	1.086551 sec	22.6% ↑
60% BTL	2.114536 sec	1.511923 sec	28.5% ↑

Table 7.1: Summary of Average Download Time Performance in FNO based Networks

The analytical model developed to represent multi-path content retrievals is evaluated by comparing the performance with the simulation model. The comparisons consider the metrics of cache hit ratios, Virtual Round Trip Time (VRTT), throughput and download times. The evaluated scenarios consist of single path, multi-path, single cache level and multiple cache levels. All the results obtained for each scenario show a close match between the analytical model and the simulation model. Therefore, from these results, it can be concluded that the analytical model is a suitable means to represent the behavior of multi-path content requests in CCN with LRU caching.

7.3 Outlook

The multi-path mechanisms and the analytical model presented in this thesis contain areas which could be further enhanced to improve the overall performance of CCN based networks. The following list details the possible areas of further improvement.

- F&CC based applications experience performance drops when using modeled caching in simulations as explained in Section 5.5.2. The reason for the performance drop is the sudden unavailability responses that a cache gives for Chunks due to the use of probabilities. A possible solution is to adopt a cache model that models at the content level rather than the Chunk level. This solution is intuitive in that, if a Chunk of a content exists already in a cache, then the other Chunks must also be present as a previous request for the content may have resulted in retrieving all the Chunks. But still, a Chunk level consideration must be made in the model due to CCN caching occurring at the Chunk level, as there is a possibility of Chunk ejections due to other content requests and due to the effects of Interest flow splitting with multi-path (i.e., cache pollution and fragmentation).

- The number of simultaneous paths that OMP-IF uses is based on research done in TCP/IP based networks ([MBNP06], [NZ01]). But in TCP/IP, there are overheads involved in establishing and using multi-path which are not present in CCN, e.g., use of extra messaging. Therefore, the overall throughput decreases when increasing the path number. Investigating the optimum number of paths to use in the context of CCN is therefore, an area to consider for future enhancements.

- The work in this thesis proposed the use of WRR and LCU as two possible path usage mechanisms and found that WRR performs better than LCU. WRR uses RTT to determine the rates of distribution of Interests to paths. When networks are congested, the fluctuations, especially the temporary fluctuations in RTT results in rapid distribution ratio changes. Therefore, an area to further investigate is the adoption of alternative mechanisms that minimize the effects of rapid changes. One such alternative is a heuristic technique which incorporates thresholds and filters [TUG+08].

- All of the Information Centric Networking (ICN) architectures have their own differences as described in Section 2.5. The work in this thesis is evaluated using the CCN architecture. Therefore, an area to further investigate is the adaptation of these mechanisms to perform equally well in other ICN architectures. Some of these mechanisms have already been adapted to work in the Network of Information (NetInf) architecture as described in [UGTG13].

Appendix

A Modeled Caching in Simulator

The analytical model that is identified in this section is used in the simulator to model caching to evaluate the CCN multi-path strategies. There are a number of reasons for using an analytical model in the simulator instead of a caching implementation that performs the actual LRU operations to save and retrieve chunks. These reasons can be summarized as follows.

- **Steady State:** Using a real LRU caching implementation has the drawback of not knowing whether the system has reached a steady state, i.e., that a cache represents the part of the content population that is considered as the most popular content. Since the CCN forwarding strategies proposed in this work uses the *Splitting Strategy* to distribute content requests to multiple paths, real growth of the caches may further be affected.
- **Resource Use:** Use of a real cache requires the use of memory and processing capabilities to store the chunks and to perform searches. This may hamper the multi-path strategy evaluations due to time and memory restrictions and further may prevent the evaluation of performance with different cache configurations.

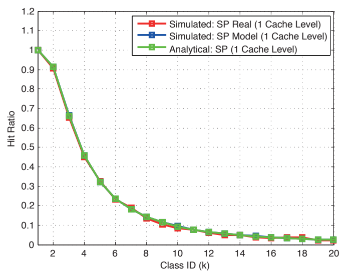
The scenarios that are considered when evaluating the forwarding strategies consist of a large number of CCN caches. Therefore, this aspect further affects the reasons mentioned above.

To verify the suitability of the analytical model in the simulator, a number of scenarios are evaluated and compared against the real LRU implementation. The main evaluation criterion is the hit ratio and in the following section the performance of the hit ratio is compared.

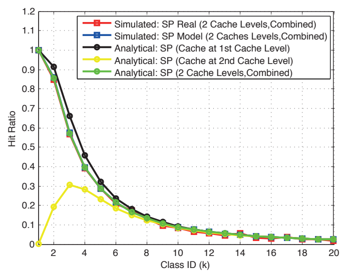
A.1 Hit Ratio

The hit ratio is computed based on the requests of chunks received at a cache against the number of chunks that were served by the cache. Figure A.1 shows the per class hit ratios for the following scenarios:

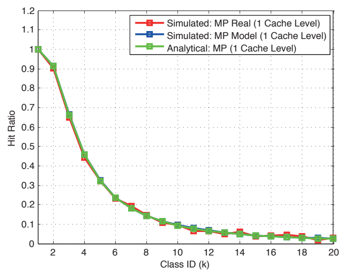
- Single path with a single and two cache level scenarios
- Multi-path with a single and two cache level scenarios



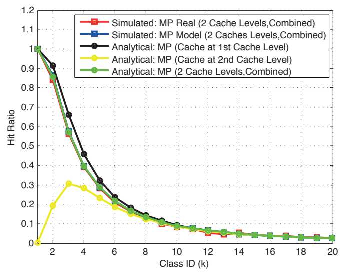
(a)



(b)



(c)



(d)

Figure A.1: Hit Ratio Comparison for Modelled and Real Cache Implementations in Simulator

This figure (A.1) compares the analytical values and the simulated performance results. The term "combined" refers to the overall performance of the caches when there is more than one cache level involved in the scenario. In the analytical case, the combined hit ratio is computed as explained in the following manner:

$h_k(i) = 1 - p_k(i)$, where $h_k(i)$ is the hit ratio of the k^{th} class at the i^{th} cache level. The number of hits generated at the i^{th} cache for the k^{th} class in the linear topology is equal to $\lambda_k \times p_k(i-1) \times h_k(i)$. The total number of hits generated at all the caches for the class k, H_k is given as in equation A.1:

$$H_k = \sum_{i=1}^{N} \lambda_k p_k(i-1) h_k(i) \tag{A.1}$$

where i represents the cache level numbers and N represents the total number of caches in the network without considering the content provider.

λ_k represents the total request arrivals at the first cache for class k. The request arrivals at the subsequent i^{th} cache in a linear topology can be given as $\lambda_k \times p_k(i-1)$. Therefore, the total requests arrivals at all the caches for the k^{th} class, TR_k is given as in equation A.2:

$$TR_k = \lambda_k + \sum_{i=2}^{N} \lambda_k p_k(i-1) \,\forall\, i > 1 \tag{A.2}$$

The overall hit ratio of the linear topology with N caches, i.e., $\frac{H_k}{TR_k}$ is computed as in equation A.3.

$$\frac{H_k}{TR_k} = \frac{\sum\limits_{i=1}^{N} p_k(i-1)h_k(i)}{1 + \sum\limits_{i=2}^{N} p_k(i-1)} \tag{A.3}$$

The results in figure A.1 shows a close match between the modeled caching in the simulator, real caching in the simulator and the analytical results.

The hit ratios for the single cache level case is shown in (a) and (c) of the figure A.1 while (b) and (d) show the results for the two level case case. (a) and (b) show the results for the single path case while (c) and (d) show the results for the multi-path case. All the curves in the single cache level case ((a) and (c)), fall on top of each other. This confirms that the model, irrespective of how the caching hit ratios are computed (i.e., mathematically or by implementing in the simulator) shows the same performance as the real LRU implementation in the simulator.

This observation is the same for the two level cache ((b) and (d)) case when considering the curves for "combined" results. The hit ratios of the first level cache is as expected as it follows the popularity of the Zipf model. But it lies above the "combined" curve at the beginning due to the second level cache dragging the hit ratio down. In the case of the second level cache hit ratio, an unorthodox behaviour is seen at the beginning of the curve. This is due to the first level cache being able to serve a high proportion of the content requests for popular content and therefore second level only being able to serve a very minor portion of these requests. In the case of the content of class 1, the graphs show that the second level cache has a hit ratio close to zero. This indicates that the second level cache can only serve a very minor portion of requests that could not be served by the first level cache.

B Statistical Estimations and Numerical Results

The purpose of this appendix is twofold. Firstly, it describes the statistical estimation methodology adopted to compute the steady-state system parameters considered in this thesis to analyze the performance of the evaluated scenarios. Secondly, it lists the numerical values of the results computed based on the selected statistical estimation methodology.

When considering the estimation of a steady-state mean value of a discrete-time simulation, the most frequently used methodologies are the Batch Means and Independent Replications. With the Batch Means, a long simulation run is split into contiguous non-overlapping batches which are then used to compute the statistical estimations. Due to the possible correlation between the batches in the Batch Means, the Independent Replications methodology is preferred.

With the Independent Replications, multiple simulations are run where each run commences from the same state and with a stream of random numbers that is different from the streams of random numbers used in the other simulation runs. The Independent Replications methodology is used in statistical estimations in this thesis. The following section provides a computation example of the Independent Replications methodology. Thereafter, in the rest of this appendix, the numerical values of the results presented in Chapters 5 and 6 are shown which were computed using the Independent Replications methodology.

B.1 Statistical Evaluation: An Example

This section provides an example of the computation of the mean and the Confidence Interval (CI) using the Independent Replications methodology. The data used for computations are collected from multiple simulation runs performed with the **FNO-BF-WCS-RCS-PCS-0BTL** scenario (tag descriptions provided in Section 5.3) and for the Average Content Object Delay metric. The CI is computed to determine the reliability and stability of the simulation results. Based on the number of replications (i.e., number of samples) performed, there are two possible alternatives to consider when computing the CI, viz., normal distribution or Student's t-distribution. A normal distribution is considered when the replication number is large while the Student's t-distribution is considered when the number of replications are small (e.g., less than 30).

The simulations consist of 10 independent simulation runs (i.e., n = 10 where n is the number of samples), each run using a different random seed. In each simulation run (i, i =

$1, 2, 3, ...n$), the Average Content Object Delay is computed as a cumulative average (T_i). Table B.1 shows the collected samples and the computations (sample mean, $\bar{T}$ and CI). Since the number of samples is less than 30, the Student's t-distribution is used. The CI for a confidence level of $(1-\alpha)$ is calculated as in Equation B.1.

$$\text{CI} = [\bar{T} - t_{(\frac{\alpha}{2},n-1)}\frac{s}{\sqrt{n}}, \bar{T} + t_{(\frac{\alpha}{2},n-1)}\frac{s}{\sqrt{n}}] \tag{B.1}$$

The value of α is set to 0.05 and $t_{(\frac{\alpha}{2},n-1)}$ can be looked up from the t-distribution table for $(n-1)$ degrees of freedom for the confidence level of $(1-\alpha)$. s is the sample standard deviation. The calculated 95% CI value for the Average Content Object Delay and its corresponding margin of errors are given in Table B.1.

Simulation run, i	**T_i, seconds**
1	0.092972
2	0.092809
3	0.092977
4	0.092833
5	0.092870
6	0.092881
7	0.093030
8	0.093042
9	0.092948
10	0.092966
Computations	**Computed Values**
Sample Mean, $\bar{T} = \frac{1}{n}\sum_{i=1}^{n} T_i$	0.092933 seconds
Sample Std. Deviation, $s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(T_i - \bar{T})^2}$	0.000079 seconds
Margin of error, $t_{(\frac{\alpha}{2},n-1)}\frac{s}{\sqrt{n}}$	0.000054 seconds
95% CI according to Equation B.1	[0.092879 , 0.092987] seconds

Table B.1: Computation of Mean and CI Values

B.2 OMP-IF Evaluation Results

The Sections 5.5.4 and 5.5.5 show the figures of the results obtained for FNO and MNO scenarios. The numerical values of theses graphs and the margins of errors are listed in this section.

The results are obtained under steady state simulations. The sample mean is estimated for all measured parameters using 10 independent simulation runs. The mean value is given with the margin of error estimated for a 95% confidence level, computed using Student's t-distribution as described in Section B.1.

B.2.1 FNO Scenario: Average Download Time and Average Content Object Delay

The mean and the margin of error values for the Average Download Time and Average Content Object Delay shown in Figure 5.11 (Page 102) are given below.

FNO: Average Download Time (sec)					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
1.062790 ±0.000819	0.900476 ±0.001984	1.401598 ±0.001028	1.082871 ±0.002840	2.113800 ±0.001140	1.511923 ±0.002984

FNO: Average Content Object Delay (sec)					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
0.092933 ±0.000054	0.070752 ±0.000080	0.124317 ±0.000049	0.086674 ±0.000082	0.192270 ±0.000097	0.123363 ±0.000132

B.2.2 FNO Scenario: Average Hop Count and Load Balancing Measure

The mean and the margin of error values for the Average Hop Count and Load Balancing Measure shown in Figure 5.12 (Page 104) are given below.

FNO: Average Hop Count					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
1.344997 ±0.004229	1.409021 ±0.003582	1.400671 ±0.004223	1.427452 ±0.003589	1.401300 ±0.012018	1.442276 ±0.010123

FNO: Load Balancing Measure					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
2.849488 ±0.006124	2.049671 ±0.007314	2.173615 ±0.003242	1.995950 ±0.003242	1.977594 ±0.002648	1.884547 ±0.003124

B.2.3 FNO Scenario: Average Retrieval Ratio

The mean and the margin of error values for the Average Retrieval Ratio shown in Figure 5.13 (Page 105) are given below.

FNO: Average Retrieval Ratio					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
0.979611 ±0.000026	0.979297 ±0.000043	0.978841 ±0.000048	0.978820 ±0.000077	0.978058 ±0.000025	0.977644 ±0.000056

B.2.4 MNO Scenario: Average Download Time and Average Content Object Delay

The mean and the margin of error values for the Average Download Time and Average Content Object Delay shown in Figure 5.20 (Page 113) are given below.

MNO: Average Download Time (sec)					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
1.414048 ±0.000210	1.388532 ±0.000810	1.852337 ±0.001569	1.681616 ±0.005249	2.943078 ±0.001656	2.548749 ±0.005468

MNO: Average Content Object Delay (sec)					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
0.122373 ±0.000098	0.111476 ±0.000113	0.161566 ±0.000112	0.142345 ±0.000448	0.253361 ±0.000115	0.211993 ±0.000516

B.2.5 MNO Scenario: Average Hop Count and Load Balancing Measure

The mean and the margin of error values for the Average Hop Count and Load Balancing Measure shown in Figure 5.21 (Page 114) are given below.

MNO: Average Hop Count					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
1.453588 ±0.004312	1.464704 ±0.00624	1.446105 ±0.004264	1.464195 ±0.004322	1.445542 ±0.014375	1.449157 ±0.013151

MNO: Load Balancing Measure					
0% BTL		**30% BTL**		**60% BTL**	
BF	**MP**	**BF**	**MP**	**BF**	**MP**
2.174029 ±0.009930	2.176827 ±0.009854	2.204852 ±0.009288	2.162495 ±0.008389	2.271456 ±0.009687	2.141467 ±0.008534

B.3 Analytical Model Evaluation Results

The Sections 6.4 and 6.5 show the figures of the results of the analytical model together with the simulated model. The numerical values of theses graphs and the margins of errors are listed in this section.

The results are obtained under steady state simulations. The sample mean is estimated for all measured parameters using 10 independent simulation runs. The mean value is given with the margin of error estimated for a 95% confidence level, computed using Student's t-distribution.

B.3.1 Hit Ratio

The mean and the margin of error values for the Hit Ratio shown in Figure 6.7 (Page 137) are given below.

Class	**Hit Ratio**		**Class**	**Hit Ratio**	
	SP	**MP**		**SP**	**MP**
1	0.999785 ± 0.000140	0.999708 ± 0.000100	2	0.907022 ± 0.000220	0.901982 ± 0.000190
3	0.652765 ± 0.000290	0.648118 ± 0.000260	4	0.448318 ± 0.000350	0.443242 ± 0.000320
5	0.325062 ± 0.000440	0.323280 ± 0.000390	6	0.232602 ± 0.000540	0.232799 ± 0.000480
7	0.189336 ± 0.000720	0.193575 ± 0.000590	8	0.134776 ± 0.000910	0.147571 ± 0.000720
9	0.102664 ± 0.000950	0.105478 ± 0.000860	10	0.084144 ± 0.001040	0.095230 ± 0.000930
11	0.075263 ± 0.001090	0.065462 ± 0.001010	12	0.060561 ± 0.001160	0.066136 ± 0.001060
13	0.050043 ± 0.001210	0.050631 ± 0.001140	14	0.047042 ± 0.001230	0.060524 ± 0.001190

15	0.036188 ± 0.001200	0.035406 ± 0.001210	16	0.032866 ± 0.001220	0.039056 ± 0.001210
17	0.036684 ± 0.001210	0.044152 ± 0.001230	18	0.036452 ± 0.001250	0.035882 ± 0.001240
19	0.021300 ± 0.000730	0.018921 ± 0.000690	20	0.021134 ± 0.000980	0.028592 ± 0.001030

B.3.2 VRTT

The mean and the margin of error values for the VRTT shown in Figure 6.9 (Page 139) are given below.

Class	**VRTT**			
	Single Level of Caches		**Two Levels of Caches**	
	SP	**MP**	**SP**	**MP**
1	2.088965 ± 0.000293	2.089193 ± 0.000209	2.089469 ± 0.000293	2.089912 ± 0.000209
2	2.283293 ± 0.000554	2.292024 ± 0.000483	2.440045 ± 0.000592	2.455446 ± 0.000517
3	2.814639 ± 0.001250	2.818183 ± 0.001131	3.309524 ± 0.001470	3.321471 ± 0.001332
4	3.241650 ± 0.002531	3.245151 ± 0.002343	4.069823 ± 0.003177	4.050257 ± 0.002924
5	3.498945 ± 0.004736	3.496549 ± 0.004218	4.589283 ± 0.006212	4.584178 ± 0.005530
6	3.692085 ± 0.008571	3.686777 ± 0.007602	5.016113 ± 0.011645	5.008851 ± 0.010328
7	3.781493 ± 0.014380	3.769272 ± 0.011488	5.218174 ± 0.019844	5.263895 ± 0.016044
8	3.896098 ± 0.026306	3.866163 ± 0.018863	5.489101 ± 0.037062	5.456178 ± 0.026621
9	3.962818 ± 0.036670	3.954932 ± 0.032246	5.668110 ± 0.052450	5.648092 ± 0.046051
10	4.001494 ± 0.049457	3.976244 ± 0.038831	5.766614 ± 0.071274	5.726724 ± 0.055926

11	4.020172 ± 0.058223	4.039064 ± 0.062318	5.841807 ± 0.084605	5.826153 ± 0.089891
12	4.050751 ± 0.077589	4.038259 ± 0.064724	5.901931 ± 0.113047	5.916472 ± 0.094827
13	4.072598 ± 0.098473	4.070782 ± 0.091657	5.980739 ± 0.144611	5.991167 ± 0.134896
14	4.078802 ± 0.106649	4.049645 ± 0.079623	5.939030 ± 0.155288	5.914641 ± 0.116292
15	4.101501 ± 0.136005	4.102888 ± 0.140215	6.047432 ± 0.200532	6.032634 ± 0.206164
16	4.108607 ± 0.152514	4.095684 ± 0.126889	6.070879 ± 0.225355	6.039789 ± 0.187119
17	4.100522 ± 0.135255	4.084805 ± 0.113796	6.038334 ± 0.199174	6.031213 ± 0.168020
18	4.101078 ± 0.140632	4.102918 ± 0.141788	6.083926 ± 0.208627	6.080360 ± 0.210124
19	4.132588 ± 0.141634	4.137325 ± 0.150879	6.115961 ± 0.209608	6.095942 ± 0.222305
20	4.132939 ± 0.191645	4.117641 ± 0.148336	6.156253 ± 0.285467	6.111686 ± 0.220171

B.3.3 Throughput

The mean and the margin of error values for the Throughput shown in Figure 6.10 (Page 140) are given below.

Class	**Throughput**			
	Single Level of Caches		**Two Levels of Caches**	
	SP	**MP**	**SP**	**MP**
1	478.705936 ± 0.067033	957.271450 ± 0.095755	478.590446 ± 0.067017	956.931838 ± 0.095721
2	437.963856 ± 0.106229	871.299394 ± 0.183537	409.828511 ± 0.099405	813.218652 ± 0.171302
3	355.285318 ± 0.157840	708.041068 ± 0.284039	302.158269 ± 0.134238	600.474492 ± 0.240887

4	308.484835 ± 0.240833	615.050759 ± 0.444037	245.710939 ± 0.191825	492.510068 ± 0.355569
5	285.800448 ± 0.386856	571.066980 ± 0.688927	217.898963 ± 0.294946	435.310941 ± 0.525153
6	270.849699 ± 0.628794	541.807936 ± 1.117134	199.357534 ± 0.462820	398.557687 ± 0.821771
7	264.445801 ± 1.005626	530.038907 ± 1.615515	191.637903 ± 0.728755	379.349749 ± 1.156227
8	256.667080 ± 1.733006	516.852584 ± 2.521728	182.179197 ± 1.230067	366.064505 ± 1.786032
9	252.345703 ± 2.335084	505.350946 ± 4.120315	176.425652 ± 1.632557	353.737756 ± 2.884156
10	249.906674 ± 3.088772	502.637479 ± 4.908652	173.411998 ± 2.143321	348.903975 ± 3.407323
11	248.745547 ± 3.602488	494.878607 ± 7.635430	171.179904 ± 2.479134	343.026906 ± 5.292526
12	246.867791 ± 4.728579	495.015531 ± 7.933918	169.436073 ± 3.245429	337.797419 ± 5.414087
13	245.543485 ± 5.937088	491.082439 ± 11.057150	167.203421 ± 4.042874	333.623154 ± 7.511817
14	245.170034 ± 6.410476	493.630163 ± 9.705615	168.377662 ± 4.402581	337.938746 ± 6.644455
15	243.813203 ± 8.084793	487.268364 ± 16.652274	165.359448 ± 5.483283	331.363829 ± 11.324276
16	243.391483 ± 9.034847	488.133760 ± 15.122925	164.720805 ± 6.114541	330.988433 ± 10.254388
17	243.871407 ± 8.044063	489.433734 ± 13.634855	165.608598 ± 5.462576	331.472547 ± 9.234305
18	243.838322 ± 8.361597	487.291481 ± 16.839715	164.367542 ± 5.636420	328.799658 ± 11.362588
19	241.979144 ± 8.293195	483.219095 ± 17.621911	163.506606 ± 5.603756	327.963056 ± 11.960073
20	241.958549 ± 11.219664	485.557007 ± 17.491976	162.436459 ± 7.532209	327.112803 ± 11.784094

B.3.4 Download Time

The mean and the margin of error values for the Download Time shown in Figure 6.11 (Page 141) are given below.

Class	**Download Time**			
	Single Level of Caches		**Two Levels of Caches**	
	SP	**MP**	**SP**	**MP**
1	0.208897 ± 0.000029	0.104464 ± 0.000010	0.208947 ± 0.000029	0.104501 ± 0.000010
2	0.228329 ± 0.000055	0.114771 ± 0.000024	0.244004 ± 0.000059	0.122968 ± 0.000026
3	0.281464 ± 0.000125	0.141235 ± 0.000057	0.330952 ± 0.000147	0.166535 ± 0.000067
4	0.324165 ± 0.000253	0.162588 ± 0.000117	0.406982 ± 0.000318	0.203042 ± 0.000147
5	0.349894 ± 0.000474	0.175111 ± 0.000211	0.458928 ± 0.000621	0.229721 ± 0.000277
6	0.369208 ± 0.000857	0.184567 ± 0.000381	0.501611 ± 0.001165	0.250905 ± 0.000517
7	0.378149 ± 0.001438	0.188665 ± 0.000575	0.521817 ± 0.001984	0.263609 ± 0.000803
8	0.389610 ± 0.002631	0.193479 ± 0.000944	0.548910 ± 0.003706	0.273176 ± 0.001333
9	0.396282 ± 0.003667	0.197882 ± 0.001613	0.566811 ± 0.005245	0.282695 ± 0.002305
10	0.400149 ± 0.004946	0.198951 ± 0.001943	0.576661 ± 0.007127	0.286612 ± 0.002799
11	0.402017 ± 0.005822	0.202070 ± 0.003118	0.584181 ± 0.008460	0.291522 ± 0.004498
12	0.405075 ± 0.007759	0.202014 ± 0.003238	0.590193 ± 0.011305	0.296035 ± 0.004745
13	0.407260 ± 0.009847	0.203632 ± 0.004585	0.598074 ± 0.014461	0.299739 ± 0.006749
14	0.407880 ± 0.010665	0.202581 ± 0.003983	0.593903 ± 0.015529	0.295912 ± 0.005818
15	0.410150 ± 0.013600	0.205226 ± 0.007014	0.604743 ± 0.020053	0.301783 ± 0.010313

16	0.410861 ± 0.015251	0.204862 ± 0.006347	0.607088 ± 0.022535	0.302125 ± 0.009360
17	0.410052 ± 0.013526	0.204318 ± 0.005692	0.603833 ± 0.019917	0.301684 ± 0.008404
18	0.410108 ± 0.014063	0.205216 ± 0.007092	0.608393 ± 0.020863	0.304137 ± 0.010510
19	0.413259 ± 0.014163	0.206945 ± 0.007547	0.611596 ± 0.020961	0.304912 ± 0.011119
20	0.413294 ± 0.019165	0.205949 ± 0.007419	0.615625 ± 0.028547	0.305705 ± 0.011013

Bibliography

[ADI^{+}11] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking (draft). In *Information-Centric Networking*, number 10492 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2011.

[APS99] M. Allman, V. Paxson, and W. Stevens. Tcp congestion control. In *IETF RFC 2581*, April 1999.

[CBD02] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing (WCMC): Special Issue on Mobile ad hoc Networking: Research, Trends and Applications*, 2:483–502, 2002.

[CBR04] I. Chakeres and E. Belding-Royer. Transparent influence of path selection in heterogeneous ad hoc networks. In *15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Barcelona, Spain*, September 2004.

[CGLD11] G. Carofiglio, M. Gallo, M. Luca, and P. Diego. Modeling data transfer in content centric networking. In *Proceedings of the 23rd International Teletraffic Congress*, San Francisco, September 2011.

[CKR^{+}07] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: analyzing the worlds largest user generated content video system. In *Proceedings of the ACM IMC*, pages 1–14, 2007.

[Col98] R. Coltun. The ospf opaque lsa option. In *IETF RFC 2370*, July 1998.

[Dan09] C. Dannewitz. Netinf: An information-centric design for the future internet. In *In Proceedings of the 3rd GI/ITG KuVS Workshop on The Future*, May 2009.

[DBM11] A. Detti and N. Blefari-Melazzi. Network layer solutions for a content-centric internet. In *Trustworthy Internet*, pages 359–369, 2011.

[DGOA10] C. Dannewitz, J. Golic, B. Ohlman, and B. Ahlgren. Secure naming for a network of information. In *INFOCOM IEEE Conference on Computer Communications Workshops*, August 2010.

[Flo04] S. Floyd. The newreno modification to tcps fast recovery algorithm. In *IETF RFC 3782*, April 2004.

[FRHB13] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. Tcp extensions for multipath operation with multiple addresses. In *IETF RFC 6824*, pages 1–14, January 2013.

[FSU$^+$07] C. Fan, M. Schlaeger, A. Udugama, V. Pangboonyanon, A. C. Toker, and G. Coskun. Managing heterogeneous access networks: Coordinated policy based decision engines for mobility management. In *In Proceedings of the 32nd IEEE Conference on Local Computer Networks, 2007 (LCN 2007)*, pages 651–660. IEEE, 2007.

[FTP11] N. Fotiou, D. Trossen, and G. Polyzos. Illustrating a publish-subscribe internet architecture. In *Journal on Telecommunication Systems, Springer*, March 2011.

[GDS$^+$03] K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file sharing workload. In *ACM SIGOPS Operating Systems Review*, volume 37, pages 314–329, 2003.

[GGSE01] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multi-path routing in wireless sensor networks. In *ACM SIGMOBILE Mobile Computing and Communications Review*, volume 5, pages 11–25, 2001.

[GR11] J. Gantz and D. Reinsel. Extracting value from chaos. *IDC iView - EMC Corporation*, June 2011.

[GXW07] J Guo, W. Xiang, and S. Wang. Reinforce networking theory with opnet simulation. In *Journal of Information Technology Education*, volume 6, pages 215–226, 2007.

[HALA07] P. Hofmann, C. An, L. Loyola, and I. Aad. Analysis of udp, tcp and voice performance in ieee 802.11b multihop networks. In *13th European Wireless Conference*, pages 1–4, 2007.

[HPSS03] O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmet. On realistic network topologies for simulation. In *In Proceedings of the ACM SIGCOMM Workshop on Models, Methods and Tools for Reproducible Network Research*, pages 28–32. ACM Press, 2003.

[JK08] P. R. Jelenkovic and X. Kang. Characterizing the miss sequence of the lru cache. In *SIGMETRICS Perform. Eval. Rev.*, volume 36, pages 119–121, August 2008.

[JST$^+$09] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging Networking Experiments and Technologies*, pages 1–12, 2009.

[KCC$^+$07] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. Kim, S. Shenker, and I. Stoica. A data-oriented (and beyond) network architecture. In *In Proceedings of SIGCOMḾ07*, August 2007.

[Kul09] K. Kuladinithi. Wireless multi-hop ad hoc networks: Evaluation of radio disjoint multi-path routing. In *Doctoral Thesis*, November 2009.

[LLP+01] R. Leung, J. Liu, E. Poon, A. Chan, and B. Li. Mp-dsr: A qos-aware multi-path dynamic source routing protocol for wireless ad-hoc networks. In *26th Annual IEEE Conference on Local Computer Networks*, pages 132–141, April 2001.

[MAK+08] Q. Mushtaq, C. An, K. Kuladinithi, A. Timm-Giel, and C. Goerg. Qos aware routing for wireless ad hoc networks. In *Baltic Conference (BaSoTi)*, August 2008.

[MBNP06] S. Mao, D. Bushmitch, S. Narayanan, and S. Panwar. Mrtp: A multiflow real-time transport protocol for ad hoc networks. In *IEEE Transactions on Multimedia*, volume 8, pages 356–369, April 2006.

[MCG11] L. Muscariello, G. Carofiglio, and M. Gallo. Bandwidth and storage sharing performance in information centric networking. In *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, pages 26–31, 2011.

[MD01] M. Marina and S. Das. On-demand multi-path distance vector routing in ad hoc networks. In *Ninth IEEE International Conference on Network Protocols*, pages 14–23, 2001.

[MGLA96] S. Murthy and J. Garcia-Luna-Aceves. Congestion-oriented shortest multi-path routing. In *INFOCOM '96, Fifteenth Annual Joint Conference of the IEEE Computer Societies, Networking the Next Generation*, volume 3, pages 1028–1036. IEEE, 1996.

[Mos14] M. Mosko. Ccnx semantics (online), July 2014. [Online; accessed 04-September-2014].

[NL07] P. Ng and S. Liew. Throughput analysis of ieee 802.11 multi-hop ad hoc networks. In *IEEE/ACM Transactions on Networking (TON)*, volume 15, pages 309–322, April 2007.

[NZ01] S. Nelakuditi and Z. Zhang. On selection of paths for multipath routing. In *IWQoS 2001*, pages 170–184, June 2001.

[PACS11] V. Paxson, M. Allman, J. Chu, and M. Sargent. Computing tcp's retransmission timer. In *IETF RFC 6298 (RFC 2988)*, June 2011.

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. In *IETF RFC 3561*, July 2003.

[PCL+11] I. Psaras, R.G. Clegg, R. Landa, W.K. Chai, and G. Pavlou. Modelling and evaluation of ccn caching trees. In *Proceedings of the 10th international IFIP TC 6 conference on Networking*, 2011.

[Pos81] J. Postel. Internet protocol. In *IETF RFC 791*, September 1981.

[PP03] P. Pham and S. Perreau. Performance analysis of reactive shortest path and multi-path routing mechanism with load balance. In *INFOCOM 2003, Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, volume 1, pages 251–259. IEEE Societies, 2003.

[RR11] D. Rossi and G. Rossini. Caching performance of content centric networks under multi-path routing (and more). In *Technical Report - Telecom ParisTech*, 2011.

[SE91] T. Stern and A. Elwalid. Analysis of separable markov-modulated rate models for information-handling systems. In *Advances in Applied Probability*, volume 23, pages 105–139, March 1991.

[Ste97] W. Stevens. Tcp slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. In *IETF RFC 2001*, January 1997.

[Tan02] A. Tanenbaum. *Computer Networks (4th Edition).* Prentice Hall, 2002.

[TUG$^+$08] U. Toseef, A. Udugama, C. Goerg, V. Pangboonyanon, and F. Pittmann. Line: Link information normalization environment. In *MOBILWARE 2008*, February 2008.

[UCG13] A. Udugama, J. Cai, and C. Goerg. Adaptation and evaluation of widely used tcp flavors in ccn. In *5th International Conference on Mobile Networks and Management, MONAMI*, September 2013.

[UGTG13] A. Udugama, C. Goerg, and A. Timm-Giel. Prototype: Ocons multi-path content delivery with netinf. In *Mobile Networks and Management*, volume 58, pages 323–327. Springer Berlin Heidelberg, 2013.

[UPG13] A. Udugama, S. Palipana, and C. Goerg. Analytical characterization of multi-path content delivery in content centric networks. *CFIC 2013*, February 2013.

[UZKG14] A. Udugama, X. Zhang, K. Kuladinithi, and C. Goerg. An on-demand multi-path interest forwarding strategy for content retrievals in ccn. *ManFI 2014*, May 2014.

[Vil99] C. Villamizar. Ospf optimized multipath (ospf-omp). In *IETF Internet Draft*, February 1999.

[VSA07] J. Vasseur, N. Shen, and R. Aggarwal. Intermediate system to intermediate system (is-is) extensions for advertising router information. In *IETF RFC 4971*, July 2007.

[Wik35] Wikipedia. Zipf's law, 1935. [Online; accessed 20-August-2014].

[Wik14] Wikipedia. Caching algorithms, 2014. [Online; accessed 20-August-2014].

[YAM$^+$13] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang. A case for stateful forwarding plane. In *Elsevier Computer Communications*, volume 36, pages 779–791, April 2013.

[YAW$^+$12] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang. Adaptive forwarding in named data networking. In *ACM SIGCOMM Computer Communication Review*, volume 42, pages 62–67, July 2012.

[YCHP08] J. Yi, E. Cizeron, S. Hamma, and B. Parrein. Simulation and performance analysis of mp-olsr for mobile ad hoc networks. *IEEE Wireless Communications and Networking Conference. WCNC 2008*, pages 2235–2240, 2008.

[YKT03] Z. Ye, S. Krishnamurthy, and S. Tripathi. A framework for reliable routing in mobile ad hoc networks. In *INFOCOM 2003 Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, volume 1, pages 270–280. IEEE Societies, 2003.

[ZAB$^+$14] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review (CCR)*, July 2014.

[Zha11] X. Zhang. Design and evaluation of multi-path strategies for content centric networking. Master's thesis, University of Bremen, January 2011.

[ZZU$^+$11] L. Zhao, Y. Zaki, A. Udugama, U. Toseef, C. Goerg, and A. Timm-Giel. Open connectivity services for future networks. *CEWIT 2011*, November 2011.