COOPERATION AND RESOURCE OPTIMIZATION IN WIRELESS NETWORKS

by

Yang Guan

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science

Summer 2014

© 2014 Yang Guan
All Rights Reserved

UMI Number: 3642314

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI 3642314

Published by ProQuest LLC (2014). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.
All rights reserved. This work is protected against
unauthorized copying under Title 17, United States Code

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

COOPERATION AND RESOURCE OPTIMIZATION IN WIRELESS NETWORKS

by

Yang Guan

Approved: _____

Errol L. Lloyd, Ph.D.
Chair of the Department of Computer and Information Sciences

Approved: _____

Babatunde Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____

James G. Richards, Ph.D.
Vice Provost for Graduate and Professional Education

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Chien-Chung Shen, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Paul D. Amer, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Errol L. Lloyd, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Leonard J. Cimini, Jr., Ph.D.
Member of dissertation committee

To my parents and to my wife.

ACKNOWLEDGEMENTS

Foremost, I would like to thank my advisor, Dr. Chien-Chung Shen, for his support, encouragement and inspiration. Over the past years, Dr. Shen has been providing so many resources and help to accommodate my research and study. Also, Dr. Shen has been constantly encouraging me to delve further into the problems at hand, as well as to keep an open mind on future research topics. Moreover, Dr. Shen's enthusiasm for research and his dedication to work have deeply influenced me since my first day at the University of Delaware. Without the mentorship and help from Dr. Shen, my Ph.D. journey could have been longer and harder.

I am also grateful to my committee members, Dr. Errol Lloyd, Dr. Paul Amer and Dr. Leonard Cimini. Dr. Lloyd helped me with several algorithm-related problems and provided precious suggestions. Dr. Amer made valuable comments on my proposal and dissertation. The collaboration with Dr. Cimini and his students made my research interdisciplinary, focusing not only on the networking issues but also on the challenges raised from the physical layer.

I feel very fortunate to work with a group of talented research colleagues, Dr. Aijun Song, Dr. Yao Xiao, Hao Feng, Dr. Justin Yackoski, Dr. Ke Li, Wei Chen, Hao Luan and Zequn Huang. We have been sharing knowledge together and helping each other with research projects. It was my privilege of working with all of them. I also greatly appreciate the assistance of our department staff, Krystal Proctor, Teresa Louise, Samantha Fowle, and Vicki Cherry, throughout my study at the University of Delaware.

I want to thank my parents for making many sacrifices that allowed me to get to this point. They have respected every decision I made, been happy for every little thing I achieved and supported me in every way they could. I could not have gone through the process without the support from my parents.

Finally, I want to thank my wife Yuqian Ding. I thank you for being here for me, for sharing my happiness and sadness, for being proud of every achievement I have made and for all the sacrifices you made for our family. Your love will always be the motivation of my work and life.

TABLE OF CONTENTS

LIST OF TABLES . . . . . **xi**
LIST OF FIGURES . . . . . **xii**
ABSTRACT . . . . . **xv**

Chapter

1 INTRODUCTION **1**

1.1 Background and Definitions 1
1.2 Problem Statement 3

1.2.1 Cooperative Source Routing in Ad hoc Networks 3
1.2.2 Transmission Scheduling for Underwater Acoustic Networks . . . 3
1.2.3 Joint Charging Tour Planning and Sensory Data Routing 4
1.2.4 Power Efficient Mobile Peer-to-Peer Streaming 5
1.2.5 Mobility-Aware Content Caching in Small-Cell Networks 6

2 COOPERATIVE SOURCE ROUTING IN AD HOC NETWORKS **7**

2.1 Introduction 7
2.2 Related Work 9
2.3 Cooperative Source Routing 10

2.3.1 Building SISO Network Topology 12
2.3.2 Cooperative RREQ Transmission 13
2.3.3 Selecting Nodes to Rebroadcast RREQs 14
2.3.4 Feedbacking RREP to Source 16
2.3.5 Transmitting Data PDUs 17

2.4 Performance Evaluation 18

2.4.1 Physical Layer Model 18
2.4.2 Simulation Parameters 19

2.4.3 Results and Discussion . . . 21
2.4.3.1 Route Discovery Success Ratio . . . 21
2.4.3.2 End-to-End Delay . . . 23
2.5 Summary . . . 23
3 TRANSMISSION SCHEDULING FOR UNDERWATER ACOUSTIC NETWORKS . . . 24
3.1 Introduction . . . 24
3.2 Related Work . . . 27
3.3 Problem Formulation . . . 28
3.3.1 Problem Definition . . . 28
3.3.2 Conflict-free Constraints . . . 29
3.3.3 Mixed Integer Linear Programming Model . . . 32
3.4 Heuristic Algorithms . . . 33
3.4.1 Conflict-Free Delay . . . 33
3.4.2 Earliest Transmission First with Random Start Scheduler . . . 34
3.4.3 Earliest Transmission First with Best Start Scheduler . . . 35
3.4.4 Naive Scheduler . . . 35
3.5 Performance Evaluation . . . 35
3.5.1 Simulation Setup . . . 35
3.5.2 Performance Comparison and Analysis . . . 36
3.6 Summary . . . 38
4 JOINT CHARGING TOUR PLANNING AND SENSORY DATA ROUTING 40
4.1 Introduction . . . 40
4.2 Related Work . . . 41
4.3 Problem Description . . . 43
4.3.1 Network Model . . . 43
4.3.2 QiF Charging Tour . . . 43
4.3.3 Distance Frequency Product . . . 44

4.3.4 The Coupling of Tour Planning and Data Routing 46

4.4 Problem Formulation 47
4.5 Heuristic Algorithm 51
4.6 Performance Evaluation 55

4.6.1 Evaluation Setup 55
4.6.2 Effects of QiF's Battery Capacity 58
4.6.3 Effects of Number of Sensors 58
4.6.4 Effects of Sensor's Transmission Range 59

4.7 Summary 59

5 POWER EFFICIENT MOBILE PEER-TO-PEER STREAMING 60

5.1 Introduction 60
5.2 Related Work 63

5.2.1 P2P Streaming to Mobile Users 63
5.2.2 Power Management in 4G LTE Networks 63

5.3 Problem Overview 66
5.4 Problem Formulation 69
5.5 Numerical Results 71
5.6 Summary 73

6 MOBILITY-AWARE CONTENT CACHING IN SMALL-CELL NETWORKS 76

6.1 Introduction 76
6.2 Related Work 78
6.3 Motivating Example 79
6.4 Mobility-Aware Content Caching Problem 80

6.4.1 Assumptions 81
6.4.2 Problem Formulation 82
6.4.3 Computational Intractability 84

6.5 Heuristic Algorithm 88
6.6 Performance Evaluation 91

6.6.1 User Mobility Model 91
6.6.2 User Preference Model 92

6.6.3 Performance of *MobiCacher* 92

6.7 Summary 99

7 FUTURE WORK 100

7.1 Scheduling of QiF Charging Tours 100

7.1.1 Research Issues and Related Work 100
7.1.2 Proposed Work 103

7.2 Dynamic Content Caching with Mobility Prediction 103

7.2.1 Predicting MU Mobility 104
7.2.2 Cache Replacement Policy 104
7.2.3 Time-Variant User Preferences 105

7.3 Content Caching at Mobile Small-Cell Base Station with Group Mobility . 106

7.3.1 Preliminary Study 107

BIBLIOGRAPHY 109

LIST OF TABLES

4.1 Definitions of constants and variables 49

5.1 Simulation parameters of P2P streaming to mobile users 72

6.1 Normalized costs of objects with respect to MU_1 and MU_2 79

LIST OF FIGURES

2.1 The format of a neighbor update PDU 11

2.2 Example of cooperative route discovery process (SISO transmission range is depicted by dashed circle; solid black arrows show the SISO transmissions of local RREQ; gray dashed arrows indicate cooperative RREQ transmissions over VMISO links; nodes with bold circle are those receiving cooperative RREQ from node A) 17

2.3 Controlled network topology 19

2.4 Success ratio of finding a route in controlled topology 20

2.5 Success ratio of finding a route in random topology 20

2.6 End-to-end delay in random topology 21

3.1 An example of transmission scheduling in UWANs [87] 26

3.2 Possible conflicts between two transmissions 30

3.3 Network throughput (number of nodes: 30; payload size: 512 bytes) . . 36

3.4 Network throughput (number of nodes: 30; payload size: 1024 bytes) . 37

3.5 Network throughput (number of nodes: 100; payload size: 512 bytes) . 37

3.6 Network throughput (number of nodes: 100; payload size: 1024 bytes) . 38

4.1 The coupling of QiF charging tours planning and sensory data routing . 45

4.2 An optimal solution for a six-node WSN 52

4.3 DFP vs. battery capacity of QiF 56

4.4 DFP vs. number of sensors in the network with different battery capacity of QiF 56

4.5 DFP vs. number of sensors in the network with different transmission ranges 57

5.1 Network infrastructure for delivering video stream to MUs 61

5.2 RRC state transitions in the 4G LTE network (a dashed arrow denotes a state transition triggered by PDU transmission/reception and a solid arrow represents a state transitions caused by timer expiration) 64

5.3 An example of sub-streams and conceptual flows 67

5.4 Maximum power consumption vs. number of MUs 73

5.5 Total power consumption vs. number of MUs 74

5.6 Maximum power consumption vs. WiFi outgoing capacity 74

5.7 Traffic over cellular links vs. WiFi outgoing capacity 75

6.1 A typical small-cell network 77

6.2 A simple SCN with two small-BSs and two MUs 79

6.3 An instance of CDP constructed from an instance of the 3-DSC problem. $T = \{a, b, c\}, S = \{\{a\}, \{c\}, \{a, b\}, \{b, c\}, \{a, b, c\}\}$. The content cached by each small-BS is shown besides each small-BS. The corresponding solution to the 3-DSC problem is $S_1 = \{\{a\}, \{b, c\}\}, S_2 = \{\{c\}, \{a, b\}\}$ and $S_3 = \{\{a, b, c\}\}$. 85

6.4 Cumulative distribution of the MUs' sojourn times 93

6.5 Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 00:00:00) 93

6.6 Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 06:00:00) 94

6.7 Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 12:00:00) 94

6.8 Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 18:00:00) 95

6.9 Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 00:00:00) 97

6.10 Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 06:00:00) 97

6.11 Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 12:00:00) 98

6.12 Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 18:00:00) 98

7.1 Executions of two QiF charging tours 102

7.2 An example of a second-order Markov mobility predictor 105

7.3 Network architecture for mobile small-BSs 106

ABSTRACT

Wireless technologies have been widely used to provide the communication functionality to various networking applications. However, the resources available to these applications are typically limited due to the fact that network devices are constrained by battery power, storage capacity and/or computational capabilities, and the fact that neighboring network devices share the common wireless spectrum with limited bandwidth. As a result, enabling cooperation among distributed network devices, and jointly optimizing the use of available resources become critical to wireless networks. In this dissertation, we study cooperation and resource optimization in various wireless networking scenarios including: ad hoc networks, underwater acoustic networks, wireless sensor networks and cellular networks. We aim to design efficient cooperation and resource optimization strategies to optimize system resource usage, and to maximize network performance.

For ad hoc networks, we study the use of cooperative communications to combat various fading effects in wireless channels. We design the Cooperative Source Routing (CSR) protocol to translate the physical-layer gain of cooperative communications into improved network-level performance. By cooperatively transmitting both route-request and route-reply control protocol data units (PDUs), CSR is capable of exploring multi-hop Virtual Multiple-Input Single-Output (VMISO) paths with high cooperative diversity.

For underwater acoustic networks (UWANs), we study the problem of transmission scheduling. We consider the issue of *Spatio-Temporal Uncertainty* attributed to the long propagation delay of acoustic signals, and specify constraints for conflict-free data transmissions and receptions in UWANs. We also design scheduling algorithms that exploit the long propagation delay of underwater acoustic signals, and facilitate concurrent transmissions and receptions to improve network throughput.

For wireless sensor networks (WSNs), we study the problem of extending the lifetime of WSNs *indefinitely*. To address this issue, we make use of the wireless energy transfer (WET) technology, and propose the paradigm of *Qi*-ferries (QiFs). QiFs are equipped with wireless energy transmitters, and periodically rove a WSN to wirelessly charge the sensors' batteries. We investigate two tightly coupled issues of how QiFs plan the charging tours to ensure that no sensor exhausts its battery, and how the sensors adjust their power consumption (by employing different routing schemes) to better accommodate the QiFs' charging activities.

For cellular networks, we consider the scenario of a group of neighboring mobile users (MUs) cooperating to download a video stream. In such a scenario, each MU requests a subset of the streaming contents over the MU's cellular link and shares the received contents with the neighboring MUs in a Peer-to-Peer (P2P) manner via a wireless local area network. We consider three issues here: the streaming data routing, the capacity of wireless local networks, and the MU's power consumption. Our objective is to determine an optimal P2P collaboration strategy that minimizes power consumption of the MUs.

Lastly, for small-cell networks (SCNs), one of the prominent challenges is a lack of sufficient backhaul capacity to connect small-cell base stations (small-BSs) to a core network. To address this challenge, we exploit the effective application layer semantics of both spatial and temporal locality to reduce the backhaul traffic, by equipping small-BSs with a storage facility to cache contents requested by users. As the cache hit ratio increases, most of the MUs' requests can be fulfilled locally without incurring traffic over the backhaul. To make intelligent caching decisions, the mobility patterns of MUs must be carefully considered as MUs might frequently migrate from one small cell to another. We thus study the mobility-aware content caching problem, with the objective to maximize the caching utility. We show this problem is NP-complete, and develop a polynomial-time heuristic algorithm with bounded approximation ratio. The performance of the heuristic algorithm is evaluated using trace-based simulations.

Chapter 1

INTRODUCTION

In this chapter we first describe the background of our research and define key technical terms. We then summarize the problems investigated in the dissertation along with the motivation for each problem. We end this chapter with the organization of the dissertation.

1.1 Background and Definitions

Wireless networking technologies have become ubiquitous. Compared to wireline networks, the resources available to wireless networks are much more limited. For example, wireless networking devices such as smart-phones and sensors are powered by batteries, which constrains both the communication and computation capabilities of these devices. Additionally, while fiber-based wireline networks offer bandwidth up to gigabits per second, the spectrum and bandwidth available to wireless communications are orders of magnitude smaller. Furthermore, neighboring wireless devices need to share a common communication spectrum due to the broadcast nature of wireless media, thus requiring careful transmission scheduling and interference avoidance to avoid excessive conflicts and collisions. Given the limited resources in wireless networks, this dissertation focuses on the cooperation among distributed networking nodes to optimize the usage of the available resources.

In this dissertation, a *protocol data unit* (PDU) refers to a unit of data exchanged between protocols in a layered networking system. A PDU contains control information (*e.g.*, header and sometimes footer) of a protocol, and possibly a service data unit (SDU) passed down by an upper layer. For example, on a transmitting device, a transport-layer PDU (T-PDU) is encapsulated as the SDU in a network-layer PDU (N-PDU). At the receiving device, the control information is removed and only the SDU portion of the N-PDU is passed to the network layer.

A *wireless ad hoc network* [77] is formed by a group of decentralized network devices, and does not rely on any pre-existing network infrastructure such as base stations or wireless hotspots. Instead, every device acts as a router that may relay and forward PDUs for other devices until the PDUs are delivered to their final destinations. The decentralized, infrastructure-less nature of the wireless ad hoc networks makes the networks suitable to provide communication capability in scenarios such as emergency rescue operations and disaster relief efforts.

A *wireless sensor network* (WSN) consists of a group of spatially distributed sensors that communicate via wireless technologies. Each sensor constantly monitors surrounding physical or environmental conditions such as temperature and humidity, and generates sensory data. The sensor's data is cooperatively relayed via other sensors towards a central location (*e.g.*, a base station or a data sink) where the sensory data is stored and analyzed. Usually, sensors in a WSN are powered by batteries, and thus have finite operational lifetime. Consequently, proper power management is a key issue to extend the lifetime of the WSN.

Underwater acoustic networks (UWANs) have emerged as the primary tool to monitor and protect the well-being of marine environments [1, 2]. Due to the high attenuation of radio-frequency (RF) signals in aquatic environments, acoustic signals are used as the communication carriers in UWANs [22]. The propagation speed of underwater acoustic signals is $\sim$1500 m/s, roughly five orders of magnitude slower than that of RF signals used by terrestrial wireless networks. Such slow propagation of underwater acoustic signals results in long propagation delays in UWANs.

Cooperative communication [57] is a physical-layer technique that combats various fading effects in wireless communications. Specifically, cooperative communication improves spectrum efficiency and link reliability by allowing properly encoded signals to be simultaneously transmitted at the source by multiple geo-distributed antennas, and correctly received at the destination.

A *cellular network* consists of many base stations distributed over a wide geographic areas. A base station covers a specific area, termed a *cell*, and is usually connected to the cellular operator's core network via a backhaul connection. The data for a mobile device within

the cell of a base station is first transported to the base station via the backhaul connection, and then transmitted to the mobile device wirelessly over certain radio frequency.

Small cell networks (SCNs) [24] refer to the networks formed by densely deployed, low-power radio access base stations including pico-cell, femto-cell and relay base stations, and WiFi access points. Compared with the traditional cellular networks, the size of each cell in SCNs is much smaller (*e.g.*, a few hundred meters in diameter [3]), which shortens the distance between a transmitter and a receiver and effectively boosts the spatial reuse and the area spectral efficiency of wireless communications.

1.2 Problem Statement

1.2.1 Cooperative Source Routing in Ad hoc Networks

Virtual Multiple-Input Single-Output is one instance of the cooperative transmission techniques capable of achieving better receiving SNR. The improved SNR can be used either to extend the transmission range or to increase the transmission data rate. To make the best use of such improvements facilitated by the cooperative communications, both routing and medium access control protocols must be adapted to benefit the performance of higher layer protocols.

In this research, we design the Cooperative Source Routing (CSR) protocol to transform the physical-layer gain of the cooperative communications into the network-level performance improvement. Specifically, CSR transmits both route request and route reply control PDUs cooperatively, and thus is capable of exploring multi-hop routes with high cooperative diversity. The performance of CSR is demonstrated through simulation and comparison with other protocols.

Chapter 2 discusses the CSR protocol in details. The work is published in [18, 84].

1.2.2 Transmission Scheduling for Underwater Acoustic Networks

Due to the long propagation delay of underwater acoustic signals, when coordinating the transmissions of PDUs in UWANs, the propagation delays between different transmitters and one receiver should be taken into account. In particular, to avoid collisions, the

signals sent by different transmitters should not overlap at the receiver in time. Moreover, in UWANs, *exclusive* access is unnecessary for collision avoidance. Instead, the successful reception of a transmitted PDU by an intended receiver must be separated in time from the receptions of other interfering signals by that receiver. This separation enables concurrent transmissions from nodes within the same carrier sense range of a receiver as long as the different propagation delays can result in different arrival times of the different signals at the receiver. Such concurrent transmissions shortens the idle periods of both transmitters and receivers, and hence boost network throughput and channel utilization.

To improve network throughput, we investigate the transmission scheduling problem in UWANs that make use of the long propagation delay of underwater acoustic signals to facilitate concurrent transmissions and receptions. We specify the constraints that resolve conflicts in UWANs, and formulate the transmission scheduling problem into a Mixed Integer Linear Programming model to calculate the optimal transmission schedule with maximum network throughput. We further design heuristics that compute conflict-free transmission schedules, and demonstrated that the heuristics can significantly improve network throughput.

The detailed study on the transmission scheduling problem in UWANs is presented in Chapter 3. The work is published in [15].

1.2.3 Joint Charging Tour Planning and Sensory Data Routing

Operating lifetime is one of the most important performance metrics of wireless sensor networks (WSNs). In certain scenarios, it is desirable to prolong the lifetime of WSNs *indefinitely*. Recent advancements of the wireless energy transfer (WET) technology [82] may facilitate such long-liveness of WSNs. Using WET, we propose to deploy *Qi*[1]*-ferries* (QiFs) equipped with energy transmitters to *periodically* rove a WSN and wirelessly charge the sensors' batteries.

Subject to the finite energy reserve on a QiF, when deploying QiFs in WSNs, two tightly coupled issues need to be addressed: (1) *charging tour planning* and (2) *sensory*

[1] Qi (pronounced "chee") means vital energy in Chinese philosophy–an intangible flow of power.

data routing. The first issue finds a set of QiF charging tours, where each tour is a cycle connecting an Energy Station (ES) with a subset of sensors in the network. The second issue of sensory data routing decides how sensory data should be relayed towards the base station[2], and thus determines the energy consumption rate of each sensor. The energy consumption rates of the sensors in turn decide how often the sensors should be visited and charged by the QiFs, *i.e.*, the frequency at which the QiFs execute their corresponding charging tours.

In this research, we explore both issues by investigating the joint charging tour planning and sensory data routing problem (JTDP). The complexity of JTDP stems from the fact that both the optimal number of QiF charging tours to be planned, and which sensors should be covered by which tours are unknown. The objective of JTDP is to minimize the *Distance Frequency Product* (DFP) of the QiF charging tours. DFP takes into account both the traveling distance of the QiF charging tours and the frequencies at which the tours are executed. As will be shown in Chapter 4, lowering DFP can effectively reduce QiFs' energy consumption on travel, thus boosting QiFs' charging efficiency.

JTDP is discussed in details in Chapter 4, and the future work on this topic is presented in Chapter 7.1.

1.2.4 Power Efficient Mobile Peer-to-Peer Streaming

In this research, we investigate how a group of neighboring mobile users (MUs) subscribing to the same video stream might cooperate with each other so as to optimize the MUs' power consumption with respect to QoS constraints. In such a cooperative scenario, in contrast to downloading the entire video stream via cellular links, each MU requests only a sub-stream of the video over the cellular link. The sub-streams are further transmitted and shared with other MUs via wireless local area networks such as WiFi. We term this scenario the *Peer-to-Peer (P2P) streaming to MUs*.

This research investigates what the most power-efficient P2P collaboration strategy is. In particular, we study the streaming data routing scheme that determines how much data an MU downloads from the cellular link, how much data an MU sends to its neighbors,

[2] Without loss of generality, we assume both energy station and base station are co-located.

and how much data an MU receives from its neighbors. The streaming data routing scheme guarantees that every MU in the network can reconstruct the original video stream, and also avoids unfairness among MUs (*i.e.*, some MUs consume much higher power than other MUs).

Chapter 5 discusses the power efficient mobile P2P streaming in details. The work is published in [16, 85].

1.2.5 Mobility-Aware Content Caching in Small-Cell Networks

Small-cell networks (SCNs) have been proposed to boost both the spatial reuse and the area spectral efficiency of wireless communications. While SCNs have the potential to provide an order-of-magnitude increase in the network capacity demanded by ever-growing mobile data traffic, challenges arise as small-cell base stations (small-BSs) are usually connected to the core network via backhauls with limited bandwidth and capacity. Additionally, the reduced cell size and large number of densely deployed small cells result in frequent migrations of MUs among cells, excessive number of handovers and rapid fluctuations in per-cell traffic load.

To address these challenges, this research investigates a solution beyond the physical layer and network topology, and exploited the application layer semantics of both spatial and temporal locality. Specifically, this work proposes to equip small-BSs with storage facilities, which cache popular contents (to be) requested by MUs. Because of the frequent migrations of MUs, this research studies the problem of which contents should be cached by each small-BS with regard to the MUs' mobility patterns. This research proves that getting an optimal solution to the problem is NP-complete, and then presents a heuristic algorithm with bounded approximation ratio.

Chapter 6 discusses the problem of mobility-aware content caching in small-cell networks in details. The work is summarized in [17], and the future work on this topic is presented in Chapter 7.2-7.3.

Chapter 2

COOPERATIVE SOURCE ROUTING IN AD HOC NETWORKS

2.1 Introduction

Space-time communications [57] have been proposed to improve spectrum efficiency and link reliability of wireless communications. To improve the performance of wireless networks, in space-time communications, properly encoded signals are simultaneously transmitted at the source and received at the destination by multi-antenna arrays.

In the context of ad hoc networks, where single-antenna nodes autonomously establish connectivity via multi-hop wireless communications, one practical approach to making use of space-time communications is to employ *virtual* antenna arrays. The elements of a virtual antenna array at the source are composed of antennas equipped on the nodes in the neighboring area, which transmit cooperatively (cooperative transmissions) to form a Virtual Multiple-Input Single-Output (VMISO[1]) system. Extensive research has exploited many physical layer coding schemes to facilitate cooperative transmissions, and the spatial diversity gain by virtual antenna arrays has been proven to approach that of multi-antenna arrays mounted on a single device [28, 66, 67]. In addition, distributed space-time coding protocol has been proposed to achieve full spatial diversity without feedback [37]. However, it remains non-trivial to effectively convert the performance gain of the VMISO at the physical layer to the performance improvement at the higher layers of the protocol stack, due to the fact that overheads are introduced to coordinate the simultaneous transmissions of different antenna elements of a virtual antenna array.

In VMISO systems, a single receiver receives different encoded copies of the same signals from multiple transmitters. When the transmission power of these transmitters is set

[1] The terms cooperative transmissions and VMISO are used synonymously.

to be the same as that of the corresponding Single-Input Single-Output (SISO) transmissions, the cooperative gain at the receiver is an increase in receiving Signal-to-Interference-and-Noise Ratio (SINR). Such a SINR increase can be used to either extend the transmission range or increase the transmission rate, or both, without increasing the bit error rate (BER). To make the best use of such improvements facilitated by the physical layer, both routing and medium access control (MAC) protocols must be adapted to benefit the performance of higher layer protocols.

Routing is essential to multi-hop ad hoc networks to discover paths between pairs of source and destination nodes. Most existing *cooperative* routing protocols designed to incorporate VMISO [28, 36] first discover a SISO route as a primary route using conventional ad hoc routing protocols such as dynamic source routing (DSR) and ad hoc on demand distance vector routing (AODV) where route request (RREQ) and route reply (RREP) control PDUs are transmitted via SISO transmissions. Then specific nodes along the primary route are selected to establish VMISO links. Thus, the quality of VMISO links highly depends on the primary SISO routes discovered. In contrast, we propose the *Cooperative Source Routing* (CSR) protocol, where both the routing request (RREQ) and routing reply (RREP) control PDUs are transmitted cooperatively. Via extensive simulation, we show that CSR can operate in network scenarios where SISO routes do not exist, and the cooperative gains can be utilized to improve network performance in terms of lower end-to-end delay and higher route discovery success ratio.

This chapter proceeds in Section 2.2 with a review of the related work on the cooperative transmissions and the cooperative routing protocols. Section 2.3 details the design of CSR, and Section 2.4 describes the performance evaluation. Section 2.5 summarizes this chapter.

2.2 Related Work

There have been several theoretical studies on the network-layer performance improvement facilitated by cooperative transmissions. In [32], the authors formulate the problem of finding a minimum energy route as a dynamic programming (DP) problem. An optimal solution to such DP problem gives rise to a multi-stage transmission power allocation scheme where minimum energy is consumed while transmitting data PDUs from a source to a destination over multiple hops. Theoretical analysis has also been conducted in this effort, and an energy saving close to 50% can be expected in arbitrary networks.

The work of [88] evaluates the benefit of cooperative transmissions in a networking context. A practical protocol termed as CoopPAC is proposed, which uses the RTS and CTS PDUs to select cooperating nodes in an uncontrolled manner. In dense networks, CoopPAC may incur extra energy consumption as CoopPAC does not limit the number nodes involved in cooperative transmissions. Also, CoopPAC only considers cooperative gain as an enhancement to the SISO transmissions, while joint optimization of transmission range extension and transmission rate increase is not discussed.

The work of [28] and [36] describe two protocols that exploit the VMISO links. Specifically, [28] emphasizes the MAC layer design to facilitate the VMISO transmissions and also proposes a routing protocol which first finds a SISO path and then shortens the SISO path through the VMISO anycasting. In [36], the authors argue that both the transmission rate and the transmission range must be jointly optimized. Specifically, [36] uses the SISO routing PDUs to collect the route information, such as the number of neighbors of each intermediate node, the level of channel interference, *etc.*, and then executes a centralized algorithm at the source to select the best route for cooperative transmissions.

We argue that the routing protocols exploiting the cooperative transmissions should not rely on an initial SISO route. Using the diversity gain facilitated by the physical layer, a cooperative routing protocol should be able to discover the VMISO routes even if there exists no SISO route between a source and a destination. In this research, we propose the CSR protocol, where both the RREQ and RREP PDUs are transmitted cooperatively over the VMISO links. In Section. 2.4, we show that CSR can survive in networks where no SISO

route is available between a source and a destination.

2.3 Cooperative Source Routing

CSR exploits VMISO links in ad hoc networks by transmitting RREQs and RREPs cooperatively. The following functions are included in CSR.

- **Building SISO network topology**: Each node periodically broadcasts a neighbor update PDU via SISO transmission. Using a process similar to distance-vector routing, the neighbor update PDUs build up a SISO network topology (SNT), where entries in SNT represent nodes that are reachable via either single-hop or multi-hop SISO transmissions.

- **Cooperative RREQ transmission**: The process of transmitting a RREQ cooperatively involves two steps: first a source node broadcasts a local RREQ to the neighbors; and then,aodvs together with the selected neighbors, the source node broadcasts a RREQ cooperatively.

- **Selecting nodes to rebroadcast RREQ**: CSR adopts a timer-based mechanism to select neighbor nodes to cooperatively rebroadcast a RREQ. The selected nodes should be close to the destination (reducing the number of hops) and have enough neighbors (exploiting cooperative diversity). Transmitting a RREQ cooperatively and selecting next-hop relay nodes proceed alternatively until RREQs arrive at the destination.

- **Feedbacking RREP to source**: Due to the asymmetry of VMISO links, a RREP is transmitted back to the source via a backward route discovery process from the destination to the source.

- **Transmitting data PDUs**: Both forward route and backward route are carried in the headers of data PDUs. Data PDUs flow along the route discovered by the RREQ to the destination and acknowledgments travel back to the source along the route discovered by the RREP.

The functions of CSR are discussed in details in Sections 2.3.1-2.3.5.

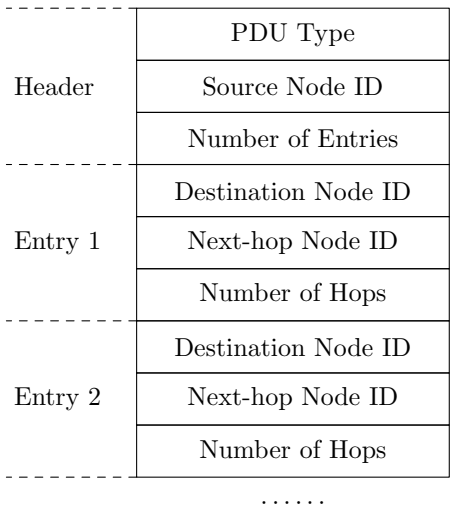


Figure 2.1: The format of a neighbor update PDU

Algorithm 1 Update SNT with an incoming neighbor update PDU

INPUT: $msg, rxNode$
$SNT \leftarrow rxNode.SNT$
if $rxNode.SNT[msg.srcNode] == NULL$ **then**
    Create entry $rxNode.SNT[msg.srcNode]$
end if
$SNT[msg.srcNode].numHops \leftarrow 1$
$SNT[msg.srcNode].nextHop \leftarrow rxNode.nodeId$
for each entry e in $msg.SNT$ **do**
    if $(SNT[e.destNode].numHops \geq e.numHops + 1)$ and
        $(e.nextHop \neq rxNode.nodeId)$ **then**
        $SNT[e.destNode].numHops \leftarrow e.numHops + 1$
        $SNT[e.destNode].nextHop \leftarrow msg.srcNode$
    end if
end for

2.3.1 Building SISO Network Topology

CSR uses a process similar to the distance-vector routing to build up a SISO network topology (SNT), where the entries in the SNT represent the nodes that are reachable via either single-hop or multi-hop SISO transmissions. Specifically, a node periodically broadcasts a *neighbor update PDU*, whose format is shown in Fig. 2.1. The *PDU Type* field indicates the PDU is a neighbor update PDU. *Source Node ID* carries the identification number of the node which initiates the PDU. The neighbor update PDU also carries a sending node's SNT information as payload, and *Number of Entries* denotes the number of entries in the node's SNT. Each entry of SNT contains three fields: *Destination Node ID*, *Next-hop Node ID* and *Number of Hops*. *Number of Hops* denotes the known minimum SISO hops from the source node to the destination node.

As all neighbor update PDUs are broadcast with SISO transmissions, *i.e.*, without the help of any other nodes, we define the nodes sending neighbor updates and the nodes receiving the corresponding neighbor updates to be neighbors. Initially, every node has an empty SNT. Upon receiving a neighbor update PDU from a neighbor node for the first time, the receiving node creates a new entry in the SNT for the node which sends the neighbor update PDU, sets the *Number of Hops* to be 1, and sets the *Next-hop Node ID* to be the receiving node. Thus, within the SNT of a node, any destination node with the *Number of Hops* being 1 is a direct neighbor of this node. Two neighboring nodes can communicate via SISO transmissions.

In addition to adding the originating node of a received neighbor update PDU into the SNT, a receiving node also updates other entries in the SNT using information piggybacked in the neighbor update PDUs. For each SNT entry in a neighbor update PDU, the receiving node checks whether an entry exists for the same destination node. If not, a new entry is created and inserted into the SNT. In an SNT entry, *Number of Hops* is the number of hops from the neighbor update PDU plus one, and *next hop node ID* is the ID of the node which sends the neighbor update. Otherwise, the receiving node updates its SNT using the

Bellman-Ford equation [34],

$$\text{numHops}_x(y) = \min_v\{1 + \text{numHops}_v(y)\}, \tag{2.1}$$

where v is the set of nodes from which node x receives neighbor update PDUs, and y is any destination node ID in the network. Algorithm 1 summarizes the process of updating SNT for an incoming neighbor update PDU.

Each entry in a SNT is time-stamped when the entry is updated. If not being updated for some time, an entry is removed from the SNT. This practice is designed to address network dynamics and node mobility.

Note that a SNT does not necessarily contain entries for all the nodes in a network, which happens when the SISO network topology is partitioned. CSR does not rely on a SNT to route data PDUs, but a SNT is used to reduce the probability of a broadcast storm of RREQ and RREP PDUs. The SNT is built in a way similar to the distance-vector routing protocol, but our implementation is light-weighted to reduce overhead. Instead of providing a complete SISO route, SNT facilitates the nodes to select the neighbors while establishing the VMISO links. Also, the SNT helps to set up the rebroadcasting timer (to be described in Section 2.3.3) and reduce the route discovery latency.

2.3.2 Cooperative RREQ Transmission

CSR discovers the VMISO links between a source and a destination by cooperatively transmitting RREQs until they reach the destination. The route discovery process is initialized by a source node which multicasts a local RREQ PDU to a subset of neighbors via SISO transmissions. The local RREQ contains the ID numbers of the source and the destination, as well as the ID numbers of the nodes which are selected to cooperate. The selection of neighbors is done in a random manner as the gain of cooperative transmissions is primarily determined by the number of cooperative transmitters [28]. Also, the local RREQ has a field carrying the source's SISO hop count to the destination. The SISO hop count is broadcast as a part of the cooperative RREQs.

After receiving a local RREQ, the selected cooperative nodes transmit pilot tones so that the potential receivers can estimate channel state information. To avoid conflicts between the pilot tones, different nodes' transmissions of the pilot tones are separated in time, in accordance with the order where the neighbors are listed in the local RREQ. The source node is the last node to transmit the pilot tone. If any selected neighbor does not receive a local RREQ or cannot transmit a pilot tone, the source can realize the fact by not hearing the corresponding the pilot tone. In this situation, the source node cancels its transmission of the pilot tone and enters a backoff stage. All selected cooperative nodes cancel further transmission accordingly if they do not hear a pilot tone from the source.

When all pilot tones are transmitted, both the source node and the selected neighboring nodes transmit a cooperative RREQ PDU. The cooperative RREQ is transmitted with no specific destination node in mind. Any node that receives an uncorrupted cooperative RREQ can respond. CSR selectively chooses nodes to rebroadcast RREQ PDUs, which is detailed next.

2.3.3 Selecting Nodes to Rebroadcast RREQs

Selecting nodes to rebroadcast the RREQs should take the physical layer capability into account. To explore a route with high cooperation diversity, a relaying node is selected based on two criteria: (1) the number of the neighboring nodes of the relay node; (2) the distance from the relay node to the destination. As indicated in [19], it is not always beneficial to transmit over many short hops, so that we demand each RREQ to go as far as possible towards the destination to reduce the number of hops.

CSR tries to minimize the coordination and communication overhead for selecting a node to rebroadcast the RREQ. To archive this goal, CSR adopts a timer-based RREQ rebroadcasting scheme. Upon a node's receiving a cooperative RREQ, if the node's SISO hop count to the destination is less than that in the received RREQ PDU, the node considers itself as a candidate and sets up a rebroadcasting timer. When the timer expires, the candidate node appends its route information (the ID number and the number of neighbors available for

cooperation) in the RREQ, and starts the cooperative transmission of the RREQ as described previously in Section 2.3.2.

The rebroadcasting timer should be directly proportional to the number of neighboring nodes, but inversely proportional to the hop count to the destination, so that the node chosen to rebroadcast the RREQ would provide high cooperative gain as well as reduce the number of hops. CSR uses a simple heuristic and calculates the value of a rebroadcasting timer as:

$$\text{Timeout} \quad = \quad (1+r) * \min\left\{\text{MAX_DELAY}, d * C * \max(1, \text{MAX_NODE} - n)\right\}, \tag{2.2}$$

where d is a candidate node's SISO hop count to the destination, and C is the maximum time to transmit one cooperative RREQ, including the time to send a local RREQ, the time to transmit pilot tones as well as the time to transmit a RREQ cooperatively. The maximum number of nodes to cooperate, MAX_NODE, is set to 8 to avoid extra energy consumption, as any further increase of the number of transmitters does not provide much more cooperative gain [36]. The MAX_DELAY is set to be the total time for a RREQ to traverse from a source to a destination with 20 cooperative RREQ rebroadcasting steps. MAX_DELAY is designed for situations where a destination does not appear in a candidate node' SNT. In other words, if the SNT does not contain an entry for the destination, the candidate node sets the timer to a large value, but still tries to rebroadcast the RREQ. This approach gives the candidate nodes which know how far the destination node is a higher priority to rebroadcast a RREQ while increasing the probability that the destination node is discovered by the RREQ. r is a random variable between 0 and 1, and is used to avoid simultaneously rebroadcasting RREQs among candidate nodes that have an identical number of neighbors and SISO hop count to the destination.

While the rebroadcasting timers are counting down, candidate nodes monitor the physical channel for incoming PDUs. Upon overhearing another node transmitting a RREQ, a candidate node cancels its rebroadcasting timer. Also, if a neighboring node transmits a

local RREQ asking a candidate node to help, the candidate node aborts its own rebroadcasting timer and cooperates. When the rebroadcasting timer expires, candidate nodes start the process of transmitting the RREQ following the steps described in the previous section. Every time a node rebroadcasts the RREQ, the node puts its ID number and the number of cooperation nodes into the RREQ. When the RREQ finally arrives at the destination, the destination knows the nodes that the RREQ has visited and the number of cooperation nodes available at each hop along the VMISO route.

Fig. 2.2 gives an example of a RREQ traveling from source A to destination R. When the upper layer of node A initiates PDUs to be sent to node R, A initializes a route discovery process by cooperatively broadcasting a RREQ with help from nodes B, C and D. The RREQ is received by nodes E, F, J, M, O and P. Node E does not try to rebroadcast because the SISO hop count of node E is larger than that of A. Nodes F, J, M, O, and P set up broadcast timers. The timer of node F expires the earliest due to F's having a smaller SISO hop count to the destination R and more neighbors. The local RREQ transmitted by F cancels the rebroadcasting timers at J and M, which together with 2 other nodes, K and L, help F transmit a cooperative RREQ to the destination. The cooperative RREQ transmitted also disables timers at O and P.

2.3.4 Feedbacking RREP to Source

After receiving the RREQ, the destination needs to feedback the discovered route to the source. During the process of finding a forward route from a source to a destination, it is possible that nodes rebroadcasting RREQ have different numbers of neighbors. Thus, the VMISO links are asymmetric. For example, in Fig. 2.2, with help from 4 neighbors, node F can transmit PDUs to node R. But node R may be unable to directly transmit to node F as node R has only two neighbors.

To address the issue of asymmetry, the destination starts an inverse route (from the destination to the source) discovery process by transmitting a route reply (RREP) PDU cooperatively. The process of inverse route discovery follows the same steps described in Section 2.3.2 and Section 2.3.3. Different from a RREQ PDU, there are payloads in the

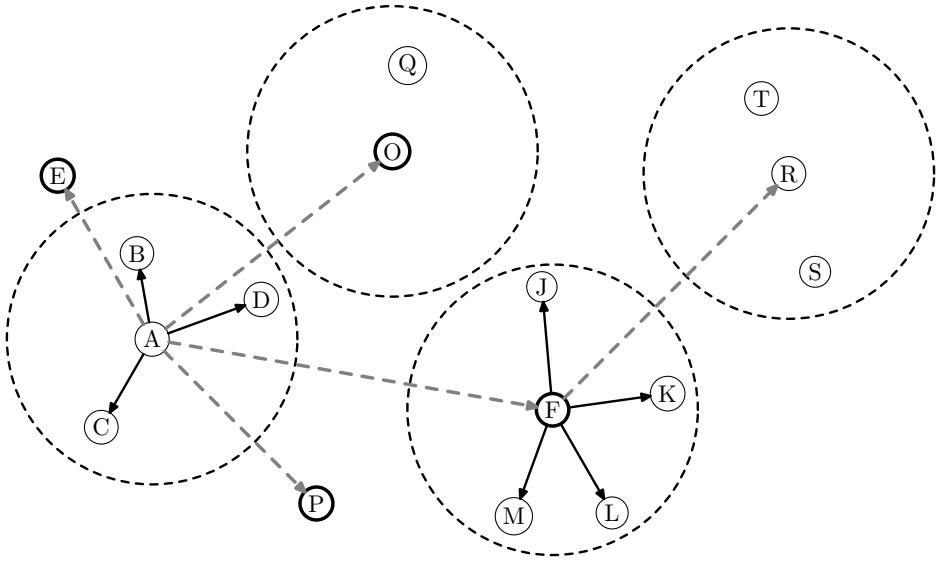


Figure 2.2: Example of cooperative route discovery process (SISO transmission range is depicted by dashed circle; solid black arrows show the SISO transmissions of local RREQ; gray dashed arrows indicate cooperative RREQ transmissions over VMISO links; nodes with bold circle are those receiving cooperative RREQ from node A)

RREP PDUs that carry the information about the discovered forward route from the source to the destination.

Upon receiving the RREP originated by the destination, the source knows both the forward route and the backward route, which enables the source to transmit data PDUs. Both the forward and backward routes are carried in data PDUs' header. The forward route guides how the data PDUs travel to the destination, and the backward route guides how the acknowledgments flow back to the source.

2.3.5 Transmitting Data PDUs

After knowing both the forward and backward routes, a source can start transmitting data PDUs. Both the forward and backward routes are carried in the headers of the data PDUs. Every time receiving a data PDU (either from the network layer or the previous relay node), a node performs a local data transmission via the SISO multicast to disseminate data to the neighbors. The neighbors used for the cooperative data transmissions are the same as those that performed the local RREQ transmission. Then, together with those neighbors, the

source starts the cooperative unicast transmission to send the data PDU to the next hop. The local data transmission and the cooperative data transmission are performed alternatively until the data PDU arrives at the destination.

Due to the asymmetry of VMISO links, CSR does not provide a hop-by-hop acknowledgment. Instead, only the destination sends an end-to-end acknowledgment to the source using the backward path specified in the data PDU header. The process of the acknowledgment flowing from the destination to the source is similar to that of the data PDUs.

Upon not receiving an acknowledgment before a certain time limit, the source retransmits the data PDU. CSR limits the number of retransmissions for a single data PDU, which causes the source node to drop the data PDU. Consecutive PDU drops indicate the route is either under interference or suffering from high outage. Thus a new route discovery process is initialized.

2.4 Performance Evaluation

We simulate CSR in QualNet 3.7 with extensions to support cooperative transmissions. In this section, we first introduce the simulation models and parameters used, and then present the simulation results.

2.4.1 Physical Layer Model

We assume all nodes transmit in the 2.4-GHz band and the channel state varies slowly such that it remains constant during the transmission time of a single PDU.

In SISO transmissions, given transmission power P_T, the receiving power at the receiver is calculated by $P_R = \frac{\alpha^2 P_T}{d^4}$, where α is the signal attenuation factor which captures both path loss and Rayleigh fading effects, and d is the distance between the sender and the receiver. In VMISO transmissions, a receiver receives several copies of the same signal transmitted simultaneously by different nodes, and the receiver calculates the total receiving power as $P_R = \sum_{i \in S} \frac{\alpha_i^2 P_T}{d_i^4}$. We use a uniform power allocation scheme where all transmitters transmit at the same power P_T.

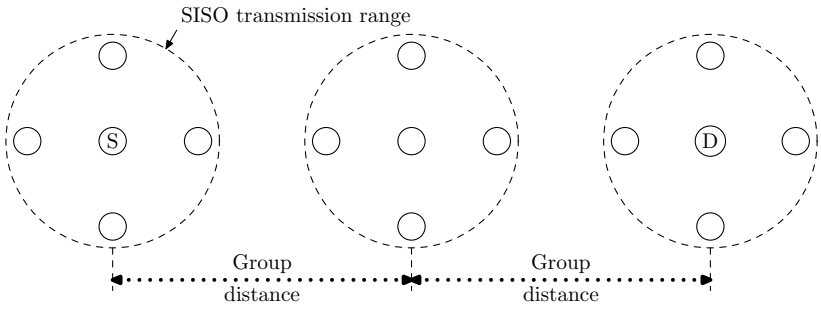


Figure 2.3: Controlled network topology

After computing the receiving power, a receiver further calculates the signal-to-interference-and-noise ratio (SINR) as SINR $= \frac{P_R}{P_\eta + P_I}$ where P_η is the noise level and P_I is the power of interfering signals. With a given SINR and modulation scheme, the bit error rate and the PDU corruption probability is computed. The receiver uses the PDU corruption probability to decide whether to drop the PDU or to deliver it to the MAC layer.

Due to the fact that transmitters are not co-located, the signals from different transmitters may arrive at the receiver at different times. It was shown in [50] that the performance of parallel relaying degrades gracefully to small delay, and the time-reversed space-time codes or the space-time OFDM can be used to combat the relative delay among different transmitters. The model tolerates $10\mu s$ difference among the arrival times of different copies of the signals. Otherwise, an incoming signal is considered as interference.

2.4.2 Simulation Parameters

Both a controlled topology and a random topology are simulated. The controlled topology is shown in Fig. 2.3, where 15 nodes are clustered into 3 groups, and the source is located in the leftmost group while the destination is located in the rightmost group. We vary the distance between groups to evaluate how CSR helps to explore routes that cannot be discovered by routing protocols with SISO transmissions. In the random topology, we use a rectangular terrain of size 1000m$\times$1000m. Nodes are randomly placed in the terrain. Also the source and the destination are selected randomly. In both controlled and random topology, a constant bit rate (CBR) application with message size of 1024 bytes is used

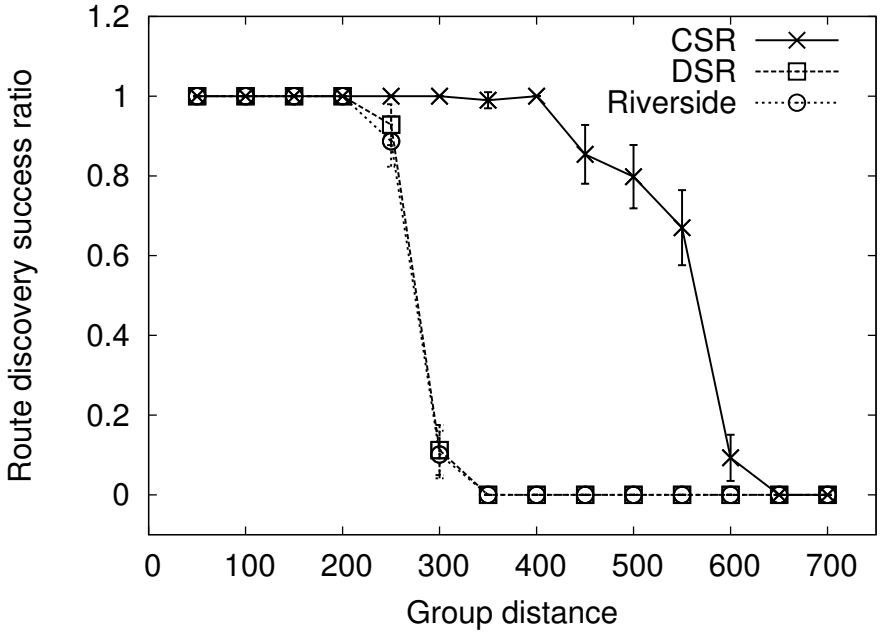


Figure 2.4: Success ratio of finding a route in controlled topology

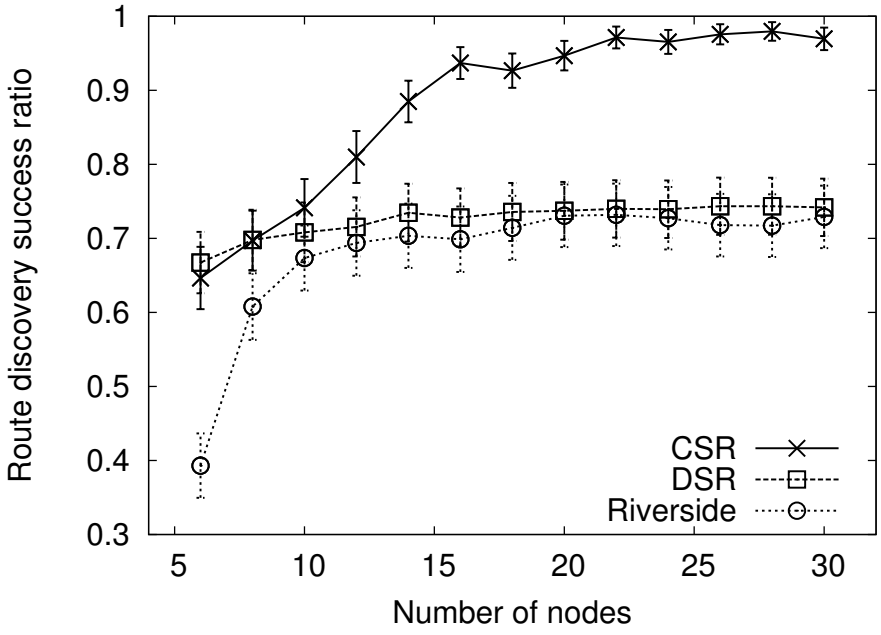


Figure 2.5: Success ratio of finding a route in random topology

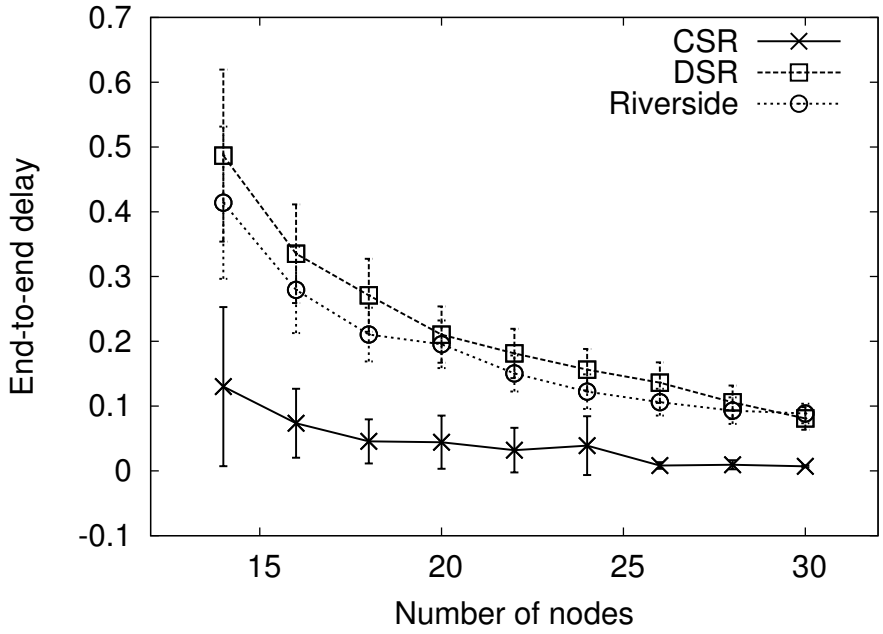


Figure 2.6: End-to-end delay in random topology

to generate traffic and UDP is the underlying transport layer protocol. Each scenario is simulated for 500 trials and the error bars in Figs. 2.4-2.6 show the 95% confidence intervals.

For comparison, we also simulate the DSR protocol and the protocol described in [28] (termed Riverside). The DSR is simulated with the standard 802.11 MAC protocol as only the SISO transmissions are involved. Riverside establishes the VMISO links based on the routes found by the DSR. We use a modified 802.11 MAC layer to support the cooperative transmissions. Riverside uses a fixed number of neighbors to cooperate with the primary nodes, which is set to 4 in our simulations.

2.4.3 Results and Discussion

2.4.3.1 Route Discovery Success Ratio

We measure the ratio at which routing protocols can find a route from a source to a destination. We apply CBR traffic to trigger a routing discovery process if a route does not exist. We limit the number of routing discovery attempts to be 20. If a route cannot be found within 20 attempts, the routing protocol reports a failure.

Fig. 2.4 shows the route discovery success ratio in the controlled network of Fig. 2.3. When the group distance is small, all three protocols show the same performance due to the fact that the nodes of the same group are within the SISO transmission range of the nodes in adjacent groups, and a route can always be found. The route discovery success ratio of both DSR and Riverside starts decreasing when the group distance is larger than the SISO transmission range and quickly reaches 0 when the group distance further increases. Notice that the ratio of Riverside is slightly lower than that of DSR. This result is due to the fact that Riverside first uses DSR to find a primary route, which may include the nodes in the same group. In this situation, the primary nodes cannot establish VMISO links as there exists insufficient number of neighbors to cooperate.

In contrast, CSR explores the benefit of the cooperative transmissions where the cooperatively transmitted RREQ and RREP can be received by the nodes in other groups even when the group distance is large (up to 400 m). This result shows that CSR is not limited by SISO transmissions, and may survive in the networks where there exists no SISO route to connect the source and the destination.

We further measure the route discovery success ratio in the random networks. All nodes are randomly deployed and both the source and the destination are also randomly chosen. Fig. 2.5 shows the results. The performance of CSR is similar to DSR when the network is sparse where most nodes do not have neighbors for cooperative transmissions and a route can only be found when there is a SISO route from a source to a destination. When the node density increases, the average degree of nodes in the network also increases, thus each node can always find neighbors to cooperatively transmit RREQ, boosting the probability of destination receiving RREQ. Riverside suffers from low route discovery ratio in sparse networks because even when the DSR discovers a primary route, the VMISO links still cannot be set up due to insufficient neighboring nodes. The route discovery success ratio of Riverside rises with the increase of the number of nodes in the network, but the performance of Riverside is still bounded by the DSR.

2.4.3.2 End-to-End Delay

We plot the average end-to-end delay of random network in Fig. 2.6. We fix the source at the bottom left of the terrain and place the destination at the top right. Any other nodes are randomly placed. We use this node placement to make the comparison fair and meaningful as different protocols may affect the route discovery success ratio. We can observe that the end-to-end delay of Riverside is smaller than that of the DSR, which is due to Riverside's path shortening mechanism that reduces the number of intermediate nodes that each data PDU visits while traveling from the source to the destination. The delay of CSR is the lowest among all three protocols. This result is because the route discovery PDUs are transmitted cooperatively and the discovered route has the potential to provide large diversity gain. The diversity gain is translated into the network-level performance in two ways. First, as the transmission range is much longer than that of the SISO transmissions, the number of intermediate nodes between a source and a destination is reduced. Second, the VMISO links boost the signal strength at the receivers, leading to the lower PDU corruption rate and reducing the number of retransmissions.

2.5 Summary

This chapter describes the cooperative source routing (CSR) protocol. CSR does not require a SISO route to be discovered first, but transmits the RREQ and RREP PDUs cooperatively, which better exploits the cooperative gain facilitated by the physical layer and the network topology. The simulation results show that CSR facilitates higher route discovery success ratio and low end-to-end delay.

Chapter 3

TRANSMISSION SCHEDULING FOR UNDERWATER ACOUSTIC NETWORKS

3.1 Introduction

Underwater acoustic networks (UWANs) have emerged as the primary tool to monitor and act upon the well-being of marine environments [1, 2]. Since radio-frequency (RF) electromagnetic signals do not propagate well in the underwater environment, UWANs use acoustic waves as the communication carriers [22], which differ fundamentally from RF signals in several ways. Foremost, the propagation speed of acoustic signals (about 1500 m/s) is roughly five orders of magnitude slower than that of RF signals. In addition, the effective transmission range of acoustic signals can reach several kilometers, which is much longer than that of RF signals [86]. Moreover, in the state-of-the-art, the bandwidth of acoustic communications is relatively low, only accommodating transmission rate of less than 60 kbps. These characteristics challenge the networking protocols of UWANs.

The major function of a Medium Access Control (MAC) protocol over a shared channel is to coordinate the transmissions from different nodes to avoid collisions. In RF wireless networks, due to the negligible propagation delay, concurrent transmissions by more than one nearby node on the same channel nearly always collide. Therefore, classical MAC protocols avoid collisions by enforcing no more than one transmitter within a receiver's carrier sense range to transmit at any time.

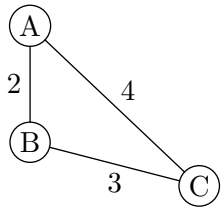
In contrast, the propagation delay of acoustic signals in UWANs is much longer, which can even exceed the transmission time of a MAC-layer data PDU. When coordinating the transmissions of data PDUs in UWANs, the distance (or propagation delay) between different transmitters and a receiver should be taken into account so that signals sent by different transmitters do not propagate to a receiver around the same time. This issue is termed the *Spatio-Temporal Uncertainty* [76].

To better demonstrate the issue of Spatio-Temporal Uncertainty, we consider the example shown in Fig 3.1. Fig. 3.1(a) depicts a UWAN topology with 3 nodes. The integer next to each link denotes the propagation delay of the link in unit of time. Figs. 3.1(b) and 3.1(c) depict two possible schedules of two transmissions (A→B and C→B), each with a transmission time of one time unit. The schedule depicted in Fig. 3.1(b), although conflict-free for terrestrial RF networks as nodes A and C do not transmit simultaneously (*i.e.*, exclusive access), results in conflict at the receiving node B in the context of UWANs as the two acoustic signals arrive at node B at the same time. In contrast, no conflict occurs in Fig. 3.1(c) even though nodes A and C transmit simultaneously. The reason is that the propagation delays interleave the arrival times of two signals at node B.

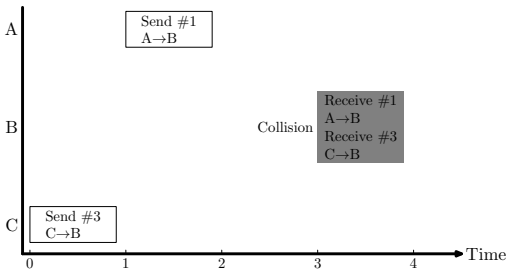
Due to the non-negligible propagation delay incurred in UWANs, exclusive access is actually unnecessary for collision avoidance. Instead, the successful reception of a transmission by the intended receiver must only be separated in time from the reception of any interfering signals by that receiver. This observation enables concurrent transmissions from nodes within the same carrier sensing range of a receiver as long as the propagation delays can result in the different arrival times of the signals at the receiver. Such concurrent transmissions can reduce the length of idle periods of both transmitters and receivers, and hence boost network throughput and channel utilization. Consider again the network in Fig. 3.1(a). The transmission schedule depicted in Fig. 3.1(d) enables conflict-free transmissions and receptions of 10 PDUs within 9 unit of time. Such performance is attributed to the fact that interference can be *stacked*. For instance, at time 5, node B simultaneously receives interference from PDU #5 sent by node C and PDU #6 sent by node A.

We can see from Fig. 3.1(d) that in UWANs, it is possible that 1) two nodes can transmit concurrently (*e.g.*, PDU #1 and PDU #2); 2) a node can transmit while receiving interference from other PDUs intended for other nodes (*e.g.*, PDU #6 is sent while node A receives interference from PDU #2); and 3) different nodes can receive different PDUs at the same time (*e.g.*, PDU #4 received by node A and PDU #3 received by node B), *all without any collisions* [87].

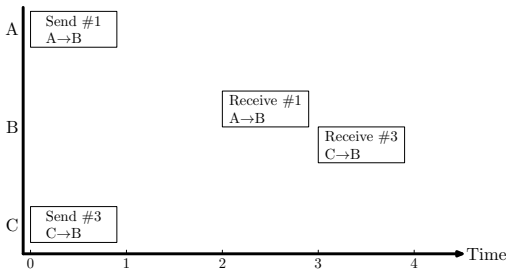
Note that such concurrent actions are impossible as a general rule in UWANs, but



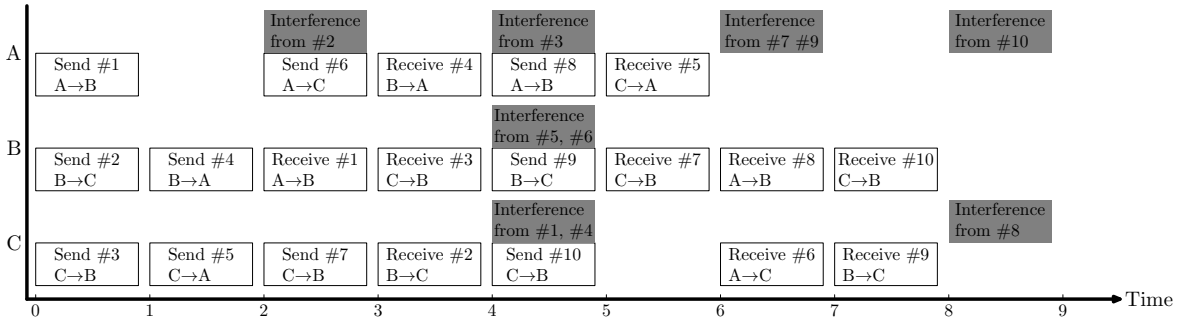
(a) Network topology with propagation delays.



(b) A schedule with transmission conflict due to Spatio-Temporal Uncertainty.



(c) A conflict-free schedule with 2 transmission tasks.



(d) A conflict-free schedule with 10 transmission tasks.

Figure 3.1: An example of transmission scheduling in UWANs [87]

are allowed in specific situations such as in Figs. 3.1(c) and 3.1(d), where the propagation delays allow nodes to receive different transmitted signals at non-conflicting times. In this research, we design scheduling algorithms that exploit the long propagation delay of acoustic signals to facilitate concurrent transmissions and receptions of acoustic communications to improve network throughput. In particular, we specify the constraints that must be satisfied to resolve transmission conflicts in UWANs, and formulate the UWAN transmission scheduling problem into a Mixed Integer Linear Programming model for optimal throughput. We also design heuristics that compute conflict-free transmission schedules and demonstrate that simple heuristics can significantly improve network throughput.

We proceed in Section 3.2 to review related work and then formulate the transmission scheduling problem in UWANs in Section 3.3. We introduce heuristics to efficiently compute the conflict-free schedules in Section 3.4, and evaluate their performance in Section 3.5. Finally, we summarize this chapter in Section 3.6.

3.2 Related Work

UW-FLASHR [87] is a distributed, time-based MAC protocol supporting isochronous traffic, where data from each user is of a constant size and is generated at a constant interval. UW-FLASHR does not require tight clock synchronization or accurate propagation delay estimation. As a time-based protocol, UW-FLASHR operates over cycles of time, where each cycle is divided into an *experimental* portion and an *established* portion. To send data, a node requests a new time slot by sending a data PDU at a random time in the experimental portion of each of several consecutive cycles. However, as each node contends to allocate a time slot by randomly choosing a transmitting time and checking to see whether such a transmission incurs any collision, UW-FLASHR gradually constructs a loose transmission schedule in a distributed manner so that idle time gaps may exist between transmissions.

ST-MAC [25] models the problem of transmission scheduling in UWANs using a weighted, directed conflict graph. ST-MAC schedules the transmissions by assigning a color (represented by an integer) to each edge in the conflict graph. Unlike the traditional vertex-coloring problem, the difference of colors between adjacent vertices must be larger than the

weight of edge between the corresponding vertices in the conflict graph. Therefore, existing heuristics cannot be easily applied to this formulation. Moreover, to make vertex-coloring algorithms work, ST-MAC has to divide transmissions into multiple unit size slots, and force the weights of edges of a conflict graph to be integers.

In our work, we consider transmission scheduling in one-hop acoustic networks, and design heuristics to solve the scheduling problem efficiently. Unlike TDMA, we do not divide a time frame into multiple equal-size time slots and a transmission can be scheduled to start at any time (instead of only at the beginning of time slots), which further improves channel utilization as well as network throughput.

3.3 Problem Formulation

In this section, we first define the transmission scheduling problem in UWANs, then present the constraints for enforcing conflict-free transmissions. Next, we formulate the transmission scheduling problem into a Mixed Integer Linear Programming (MILP) model for optimal throughput. Since the computation time needed to calculate optimal solutions for large networks is prohibitively long, we run an MILP solver for small networks and use the results as benchmark to evaluate heuristic solutions presented in Section 3.4.

3.3.1 Problem Definition

To define the transmission scheduling problem for UWANs, we assume the existence of a base station which collects information about all of the transmission tasks and the propagation delays among nodes, and computes the transmission schedules for each task. We model a UWAN as a directed graph $G = (V, E)$, where each vertex $v \in V$ represents a node in the network capable of transmitting and receiving the acoustic signals. Each $e \in E$ represents an acoustic link between two nodes. In this work, we consider an one-hop acoustic network in which each node can hear the signal from any other node, *i.e.*, G is fully connected[1]. A weight function W maps each edge in E into a real number representing the

[1] If G is not fully connected, a multi-hop transmission can be considered as several single-hop transmissions, and the conflict-free constraints between two single-hop transmissions can be properly specified based on whether one transmission is within the interference range of the other transmission.

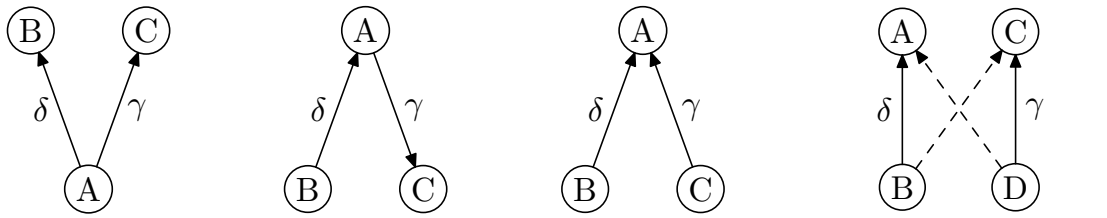
propagation delay, which is calculated by dividing the distance between the corresponding transmitter and receiver by acoustic propagation speed. We notice that the path along which acoustic signal propagates can be affected by factors such as salinity and temperature of water, and the time an acoustic signal travels from one node to another might unnecessarily be equal to the time that the acoustic signal travels in the reverse direction. We address this issue by adding guard time between adjacent transmissions.

The transmission scheduling problem in UWANs concerns a set Δ of *transmission tasks*. Each task $\delta \in \Delta$ is specified by the *source* (δ_s), the *destination* (δ_d), the *transmission duration* (δ_{tx}), and the *start transmitting time* (δ_τ). The first three items of each transmission task are known *a priori*. The goal of the scheduling problem is to compute the proper *start transmitting time* for each transmission task so as to avoid conflicts and maximize network throughput. The base station collects the information about the source, the destination and the transmission duration of each transmission task, and uses G, W and Δ as input to the scheduling algorithm which computes the start transmitting time for each transmission task. The schedules computed are assumed to be non-preemptive, so that the scheduling algorithms do not split transmission tasks into pieces.

3.3.2 Conflict-free Constraints

As discussed previously, in UWANs, two nodes within the carrier sensing range of a receiver can transmit at the same time without causing conflict as long as the arriving signals do not overlap at the receiver. We aim to design algorithms that compute the transmission schedule such that all conflicts at the receivers can be avoided. We call such a conflict-free schedule *valid*. In the remainder of this subsection, we list the conflict-free constraints that a valid schedule needs to satisfy.

To derive conflict-free constraints, we first discuss how two transmission tasks (denoted as δ and γ) may conflict with each other, and term the node(s) involved as *conflicting node(s)*. Based on whether a conflicting node is a transmission source or a transmission destination, we divide conflicts into four categories: (a) *TX-TX* conflict (Fig. 3.2(a)); (b)



(a) TX-TX conflict (b) TX-RX conflict (c) RX-RX conflict (d) RX-Interference conflict

Figure 3.2: Possible conflicts between two transmissions

TX-RX conflict (Fig. 3.2(b)); (c) *RX-RX* conflict (Fig. 3.2(c)) and (d) *RX-Interference* conflict (Fig. 3.2(d)). It is worth pointing out that interference signals do not cause conflicts when the signals propagate to nodes that are transmitting. We first enumerate constraints for conflicts in each category. We then show that all the categories can be treated in a unified manner, and give two inequality constraints to avoid all of the possible conflicts.

(a) *TX-TX* conflict. Since each node is assumed to be equipped with only one transceiver, a node cannot transmit more than one data PDU at the same time. *TX-TX* conflict occurs when two transmission tasks share the same source node ($\delta_s = \gamma_s$) but have different destinations (transmission tasks with a common source and a common destination can be aggregated and considered as one single transmission). In *TX-TX* conflict, the conflicting node is the common source node. For two transmissions tasks sharing one common source, the tasks must be interleaved so that either (3.1) or (3.2) is to be satisfied. Eq. (3.1) considers the case that δ is transmitted before γ and (3.2) takes care of γ being transmitted before δ.

$$\delta_\tau + \delta_{tx} \leq \gamma_\tau, \tag{3.1}$$

$$\gamma_\tau + \gamma_{tx} \leq \delta_\tau. \tag{3.2}$$

(b) *TX-RX* conflict. A node cannot receive signals destined to itself while transmitting another signal at the same time. A valid schedule guarantees that an incoming data

PDU either arrives at a conflicting node after the conflicting node finishes transmitting (3.3) or ends before the conflicting node starts transmitting (3.4):

$$\delta_\tau + \delta_{tx} + W_{(<\delta_s,\delta_d>)} \leq \gamma_\tau, \tag{3.3}$$

$$\delta_\tau + W_{(<\delta_s,\delta_d>)} \geq \gamma_\tau + \gamma_{tx}, \tag{3.4}$$

where $W_{(<v,u>)}$ denotes the propagation delay from node v to node u.

(c) *RX-RX* conflict. A node cannot receive more than one data PDU at a time. In *RX-RX* conflict, two transmissions share one common destination node, which is the conflicting node. Different from *TX-TX* conflict, a valid schedule needs to take the propagation delay into consideration. For example, in Fig. 3.2c, both δ and γ have data destined to node A. When scheduling the start transmission times, the propagation delay from B to A and the propagation delay from C to A should be considered. In general, to resolve *RX-RX* conflicts between δ and γ, one of the following two inequality equations must be satisfied:

$$\delta_\tau + \delta_{tx} + W_{(<\delta_s,\delta_d>)} \leq \gamma_\tau + W_{(<\gamma_s,\gamma_d>)}, \tag{3.5}$$

$$\delta_\tau + W_{(<\delta_s,\delta_d>)} \geq \gamma_\tau + \gamma_{tx} + W_{(<\gamma_s,\gamma_d>)}. \tag{3.6}$$

Eqs. (3.5) and (3.6) resolve *RX-RX* conflicts at δ_d. The corresponding constraints to resolve conflicts at γ_d are similar, and hence are omitted.

(d) *RX-Interference* conflict. The signal from one transmission may interfere with the signal of another transmission at the latter transmission's intended destination. Different from the previous conflict scenarios where transmissions share nodes as either the source or the destination, here two transmissions are disjoint and both destinations are conflicting nodes. We must schedule the start transmitting times of concurrent transmission tasks such that interference signals do not arrive at the conflicting nodes when they are receiving legitimate signals. The constraints to resolve *RX-Interference*

conflicts are the same as those for *RX-RX* conflicts.

By examining (3.1)–(3.6), we find that (3.5) is actually the generalized form of (3.1) and (3.3), and (3.6) is a generalized form of (3.2), and (3.4), if we define the propagation delay from one node to itself to be zero, *i.e.*, $\forall v \in V, W_{(<v,v>)} = 0$. This observation provides us an easy way to formulate the scheduling problem into an MILP problem.

3.3.3 Mixed Integer Linear Programming Model

Before formulating the transmission scheduling problem into a MILP model, we notice that popular MILP solvers such as GLPK [48] do not support disjunction form of constraints as (3.5) and (3.6). We overcome this issue by introducing a binary parameter $b_{\delta\gamma}$ and rewriting the constraints as:

$$\delta_\tau + \delta_{tx} + W_{(<\delta_s,\delta_d>)} \leq \gamma_\tau + W_{(<\gamma_s,\delta_d>)} + b_{\delta\gamma} * C, \tag{3.7}$$

$$\delta_\tau + W_{(<\delta_s,\delta_d>)} \geq \gamma_\tau + \gamma_{tx} + W_{(<\gamma_s,\delta_d>)} + (b_{\delta\gamma} - 1) * C, \tag{3.8}$$

where C[2]is a constant that is large enough to guarantee when $b_{\delta\gamma} = 0$, (3.7) is automatically satisfied and when $b_{\delta\gamma} = 1$, (3.8) is true all the time. $b_{\delta\gamma}$ actually defines the order among transmissions. For example, when $b_{\delta\gamma} = 1$, the conflicting node will serve δ before γ.

We also need to re-define the term *frame size* ($\mathcal{F}$). In terrestrial RF networks, frame size is usually set to be the number of transmission slots in one epoch. In UWANs, due to the large propagation delays, a new frame cannot begin immediately after the last transmission finishes otherwise inter-frame conflicts may happen. One method to avoid inter-frame conflicts is to append large guard time (up to the maximum propagation delay in the network) at the end of each frame to make sure the channel become idle when the next frame starts. This practice, however, reduces network throughput since the guard period needs to be as long as the maximum propagation delay. We define the *frame size* ($\mathcal{F}$) as the minimum delay between two successive transmissions to avoid inter-frame conflicts, and try to minimize

[2] $C > (|\Delta| + 1)(T_{max} + P_{max})$ where $|\Delta|$ is the total number of transmissions, T_{max} is the maximum transmission delay and P_{max} is the maximum propagation delay.

the frame size since it is inversely proportional to the network throughput. The constraint to resolve inter-frame conflicts is:

$$\gamma_\tau + \gamma_{tx} + W_{(<\gamma_s,\delta_d>)} \leq \delta_\tau + \mathcal{F} + W_{(<\delta_s,\delta_d>)} \tag{3.9}$$

We formulate the scheduling problem into MILP as follows:

$$\min \quad \mathcal{F} \tag{3.10}$$

$$\text{s.t.} \quad \delta_\tau + \delta_{tx} + W_{(<\delta_s,\delta_d>)} \leq \gamma_\tau + W_{(<\gamma_s,\delta_d>)} + b_{\delta\gamma} * C, \qquad \forall \delta, \gamma \in \Delta, \tag{3.11}$$

$$\delta_\tau + \delta_{tx} + W_{(<\delta_s,\gamma_d>)} \leq \gamma_\tau + W_{(<\gamma_s,\gamma_d>)} + b_{\delta\gamma} * C, \qquad \forall \delta, \gamma \in \Delta, \tag{3.12}$$

$$\delta_\tau + W_{(<\delta_s,\delta_d>)} \geq \gamma_\tau + \gamma_{tx} + W_{(<\gamma_s,\delta_d>)} + (b_{\delta\gamma} - 1) * C, \qquad \forall \delta, \gamma \in \Delta, \tag{3.13}$$

$$\gamma_\tau + \gamma_{tx} + W_{(<\gamma_s,\delta_d>)} \leq \delta_\tau + \mathcal{F} + W_{(<\delta_s,\delta_d>)} b_{\delta\gamma} * C, \qquad \forall \delta, \gamma \in \Delta, \tag{3.14}$$

$$\delta_\tau + \delta_{tx} + W_{(<\delta_s,\delta_d>)} < \gamma_\tau + W_{(<\gamma_s,\delta_d>)} + b_{ij} * C, \qquad \forall \delta, \gamma \in \Delta, \tag{3.15}$$

$$\gamma_\tau + \gamma_{tx} + W_{(<\gamma_s,\gamma_d>)} \leq \delta_\tau + \mathcal{F} + W_{(<\delta_s,\gamma_d>)} b_{\delta\gamma} * C \qquad \forall \delta, \gamma \in \Delta. \tag{3.16}$$

3.4 Heuristic Algorithms

Although the MILP model computes optimal schedules to archive maximum throughput, the practical computation time taken to obtain such optimal schedules is quite long (*e.g.*, it takes about 350 seconds to compute the optimal transmission schedule of 11 transmissions in a 30-node UWAN). In this section, we introduce two heuristic solutions for the UWAN transmission scheduling problem. With these heuristic schedulers, we can efficiently compute conflict-free schedules for large networks with complex traffic patterns. As shown in Section 3.5, these two simple heuristics demonstrate similar performance to the optimal schedule.

3.4.1 Conflict-Free Delay

To better explain our heuristics, we first define the *conflict-free delay* (CFD) of transmission task γ after transmission task δ (denoted as $\mathcal{C}_{\delta\to\gamma}$) as the minimum time γ must wait after δ starts transmitting to avoid conflicts. If γ starts transmitting earlier than the

conflict-free delay, conflicts will occur. From the discussion above, $\mathcal{C}_{\delta\to\gamma}$ is formally defined as follows,

$$\mathcal{C}_{\delta\to\gamma} = \max\{\delta_{tx} + W_{(<\delta_s,\delta_d>)} - W_{(<\gamma_s,\delta_d>)}, \delta_{tx} + W_{(<\delta_s,\gamma_d>)} - W_{(<\gamma_s,\gamma_d>)}\} \tag{3.17}$$

3.4.2 Earliest Transmission First with Random Start Scheduler

The first heuristic scheduler is termed *Earliest Transmission First with Random Start* (ETFRS), shown in Algorithm 2. ETFRS initializes all transmission tasks as unscheduled and sets their start transmitting time to be 0 (Lines 1-2). ETFRS then randomly selects one transmission task as the first task to be scheduled (line 3-4). Each time a transmission task is scheduled, ETFRS updates the unscheduled tasks' potential start transmitting time to avoid conflicts between the unscheduled tasks and the scheduled tasks (Lines 7-11). After the update is finished, ETFRS selects an unscheduled transmission task with the minimum potential start transmitting time and schedules this transmission task next (Lines 12-13). The process terminates when all the transmission tasks have been assigned with the start transmitting time. Finally, the frame size is calculated as the minimum time between the two successive transmissions of the same task (Line 15).

Algorithm 2 Earliest Transmission First with Random Start

1: $\forall\delta \in \Delta, \delta_\tau \leftarrow 0$
2: $\forall\delta \in \Delta$, SCHEDULED$[\delta] \leftarrow$ FALSE
3: Randomly select a transmission task $\delta \in \Delta$
4: $\theta \leftarrow \delta$
5: **while** Exists unscheduled tasks **do**
6: SCHEDULED$[\theta] \leftarrow$ TRUE
7: **for** $(\forall\gamma \in \Delta)$ and (SCHEDULED$[\gamma] ==$ FALSE) **do**
8: **if** $\gamma_\tau < \theta_\tau + \mathcal{C}_{\theta\to\gamma}$ **then**
9: $\gamma_\tau \leftarrow \theta_\tau + \mathcal{C}_{\theta\to\gamma}$
10: **end if**
11: **end for**
12: Select an unscheduled task λ with minimum λ_τ
13: $\theta \leftarrow \lambda$
14: **end while**
15: $\mathcal{F} = \max_{\theta\in\Delta,\theta\neq\delta}\{\alpha_\tau + \mathcal{C}_{\alpha\to\delta}\}$

3.4.3 Earliest Transmission First with Best Start Scheduler

The second heuristic scheduler, termed *Earliest Transmission First with Best Start* (ETFBS), is similar to ETFRS. However, instead of randomly selecting a transmission task as the first task to be scheduled, ETFBS calls Algorithm 2 with a different starting task and finds a schedule with the minimum frame size. In other words, ETFBS selects the best schedule out of $|\Delta|$ conflict-free schedules.

3.4.4 Naive Scheduler

The naive scheduler simply schedules a sequence of transmission tasks in a conflict-free manner, and does not alternate the order of transmission tasks to exploit concurrent transmissions and receptions. We run naive scheduler as a baseline to show how concurrent transmission tasks can improve network throughput.

3.5 Performance Evaluation

In this section, we describe the simulation setup used to evaluate the two heuristic and the naive schedulers. We run simulations with different networking scenarios and compare the performance of different heuristics against the optimal scheduling computed by the MILP model.

3.5.1 Simulation Setup

We use QualNet 3.7 [62] to evaluate the performance of different heuristics. To simulate acoustic channels, we extended QualNet with spherical path loss and Thorp attenuation. We use a terrain of 1000m $\times$ 1000m and randomly place 30 nodes (for small networks) or 100 nodes (for large networks). As mentioned above, the transmission range of acoustic signals can reach several kilometers, so in our simulated networks, all nodes can hear each other. The propagation speed is 1500 m/s and the transmission rate is 15 kbps. For small networks, we simulate up to 10 transmissions with random source and destination pairs, which allows us to run GLPK [48] to solve the MILP and obtain the optimal transmission schedule within reasonable time. For large networks, up to 30 transmissions are generated and transmission

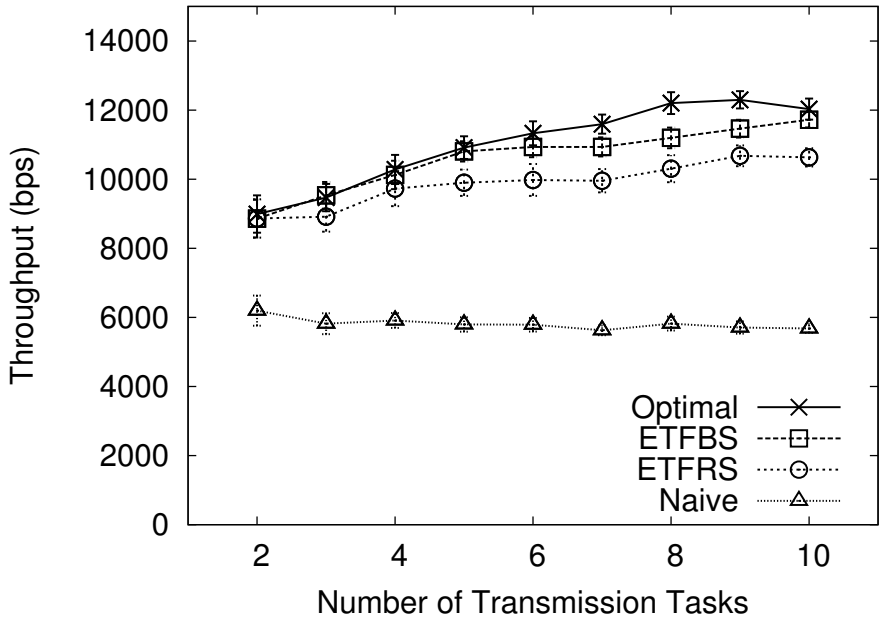


Figure 3.3: Network throughput (number of nodes: 30; payload size: 512 bytes)

schedules are computed from heuristics described above. The transmissions are generated as a Constant Bit Rate (CBR) application with payload size of either 512 bytes or 1024 bytes. Each network configuration (number of nodes, number of transmissions and payload size) is simulated 40 times with different node placements and different source-destination pairs to obtain the average throughput.

3.5.2 Performance Comparison and Analysis

Figs. 3.3-3.6 show the results of network throughput. We can see from Figs. 3.3 and 3.4 that the optimal schedule obtained by solving the MILP model archives the highest throughput. The throughput of the optimal schedule increases with the number of transmissions, due to the fact that when there are only a few transmissions, it is difficult to schedule conflict-free concurrent transmissions, so that most nodes will remain idle when the signals are propagating. However, the throughput of the optimal schedule is still 32% higher than that of the naive schedule, mainly due to the fact that the naive scheduling does not take advantage of the Spatio-Temporal Uncertainty, and ill-scheduled start transmitting times reduce the channel efficiency.

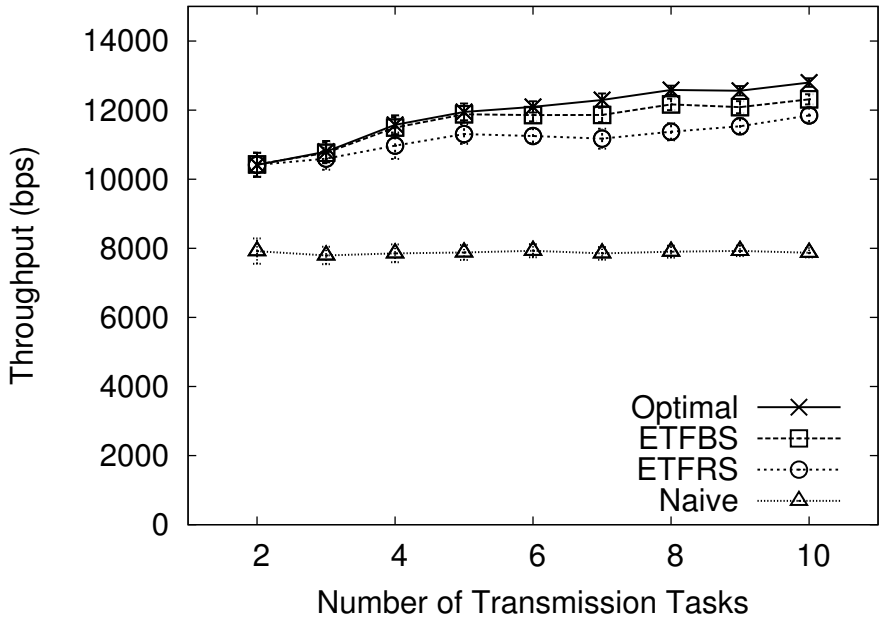


Figure 3.4: Network throughput (number of nodes: 30; payload size: 1024 bytes)

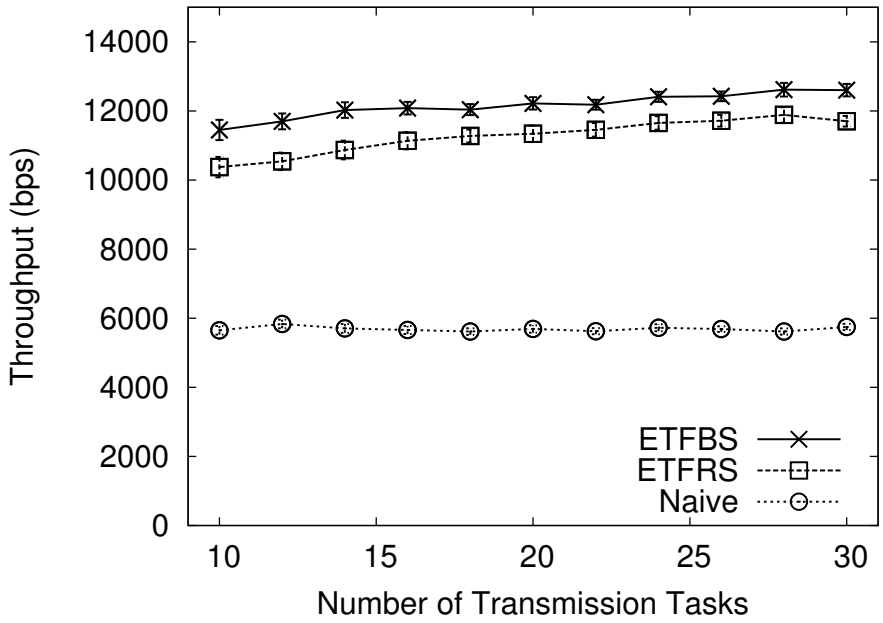


Figure 3.5: Network throughput (number of nodes: 100; payload size: 512 bytes)

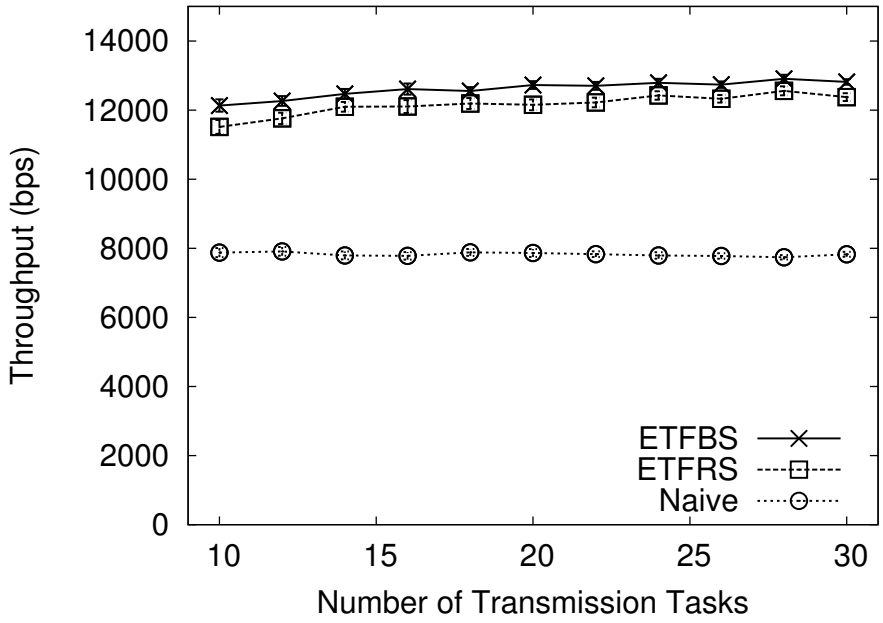


Figure 3.6: Network throughput (number of nodes: 100; payload size: 1024 bytes)

Comparing Fig. 3.3 and Fig. 3.4, we can see that transmissions with large payload (1024 bytes) results in higher throughput than that with small payload (512 bytes). The reason is that the transmission time of larger frames is longer, while the propagation delay remains the same, so that the issue of Spatio-Temporal Uncertainty becomes less significant. When frames with megabyte payload are transmitted, the propagation delay becomes negligible compared with transmission delay, and we come back to a situation that is similar to terrestrial networks. Figs. 3.5 and 3.6 confirm this point. The ETFBS heuristic achieves throughput close to the optimal schedule. This fact makes ETFBS an ideal candidate to schedule large scale UWANs and/or UWANs with heavy traffic, as the time complexity of ETFBS is $O(|\Delta|^3)$ and the practical computation time of ETFBS is much less than the time to obtain an optimal schedule via an MILP solver.

3.6 Summary

This research investigates the problem of transmission scheduling in UWANs that takes advantage of the Spatio-Temporal Uncertainty to exploit concurrent, conflict-free transmissions and receptions of acoustic communications. A Mixed Integer Linear Programming

model is formulated to compute the optimal transmission schedules for the maximum network throughput. Also, heuristic algorithms are designed to efficiently compute the conflict-free schedules over large networks. The simulation results show that the heuristics can significantly improve network throughput over a naive scheduler.

Chapter 4

JOINT CHARGING TOUR PLANNING AND SENSORY DATA ROUTING

4.1 Introduction

In wireless sensor networks (WSNs), sensor nodes are typically powered by batteries. While the lifetime of a WSN can be prolonged by lowering sensors' energy consumption rates, it is desirable to extend its lifetime *indefinitely*. Although energy (such as solar and wind) may be harvested from the environment to replenish sensors' batteries [31], such energy may be unstable or unpredictable.

Recent advances in wireless energy transfer (WET) technology [82] may facilitate the long-liveness of WSNs. With WET, a sensor node can receive energy sent by an energy transmitter without any physical cord. The feasibility of wirelessly charging sensors' batteries has been demonstrated in field experiments with commercial WET products [78], and the loss of energy is reasonable even when the transmitter and the receiver are separated by a distance (*e.g.*, the transfer efficiency is more than 85% when the wireless energy transfer distance is less than 1 meter [35]).

Using WET, we propose to deploy *Qi*[1]*-ferries* (QiFs) equipped with energy transmitters to periodically rove a WSN and wirelessly charge the sensors' batteries. As the movement of a QiF is also powered by electric energy[2], to optimize the usage of the entire energy reserve on a QiF, both the movement of a QiF and the wireless charging of sensor nodes share the same pool of battery energy, which results in a clear tradeoff between how many sensors a QiF could charge and how far the QiF could travel.

---

[1] Qi (pronounced "chee") means vital energy in Chinese philosophy–an intangible flow of power.

[2] QiFs are designed to be eco-friendly without burning any fossil fuel.

Subject to the finite energy reserve on a QiF, when deploying QiFs in WSNs, two tightly coupled issues need to be addressed: (1) *QiF charging tour planning*, and (2) *sensory data routing*. The first issue finds a set of QiF charging tours, where each tour is a cycle connecting an Energy Station (ES) with a subset of sensors in the network. Periodically, a QiF departs from the ES, travels along a QiF charging tour to charge sensors covered by the tour, and returns to the ES with non-negative residual energy. Given the finite energy reserve on a QiF, multiple QiF charging tours may be needed to cover all of the sensors in a large scale WSN. The second issue of sensory data routing decides how sensory data should be relayed towards the base station[3], and thus determines the energy consumption rate of each sensor. The energy consumption rates of the sensors in turn decide how often the sensors should be visited and charged by QiFs, *i.e.*, the frequency at which QiFs carry out the corresponding charging tours.

In this research, we propose to jointly explore both issues by investigating the joint charging tour planning and sensory data routing problem (JTDP). The remainder of this chapter is organized as follows. The next section reviews related work and Section 4.3 describes JTDP in details. In Section 4.4, JTDP is formulated into a Mixed Integer Linear Programming (MILP) problem for optimal solution. Since solving a MILP problem is NP-hard, we propose a heuristic solution in Section 4.5 to plan charging tours and decide data routing for large scale WSNs. Performance of the heuristic solution is evaluated in Section 4.6 and Section 4.7 summarizes this research.

4.2 Related Work

There have been a few efforts on applying WET to WSNs. In [78], the authors discuss how to deploy sensors at each post of interest, and how to route the sensory data towards the base station with the objective of increasing the wireless charging efficiency. [69] plans the activities of wireless charging vehicle (WCV) to charge sensors in WSNs with random topologies. The authors show that the optimal approach for a WCV to charge sensors in a WSN is to travel along the shortest Hamiltonian cycle connecting all sensors with an ES. In

[3] Without loss of generality, we assume both energy station and base station are co-located.

addition, the authors develop a near-optimal algorithm to calculate the data routing and the charging time for each sensor so as to increase the portion of WCV's vacation time (*i.e.*, the time that WCV stays at the ES). However, [69] does not consider the limited energy reserve on a WCV by assuming that, once departing from the ES, a WCV always has enough energy to charging all the sensors in the network. Also, as sensors may consume energy at drastically different rates, the frequency at which sensors are charged is determined by the *bottleneck* sensor that consumes energy at the highest rate. If such an issue is not considered, some sensors may be charged prematurely when their battery levels are still high.

In this research, we plan to investigate a more practical scenario in which each QiF has a finite energy reserve, and QiF's energy consumption for both movement and wireless charging of sensors is taken into account. This scenario is in sharp contrast to [42] which assumes that the movement of a mobile charger consumes zero energy. Given the finite capacity of QiF's energy reserve, a solution to JTDP may include multiple QiF charging tours to cover sensors in a WSN, with each sensor covered by exactly one tour. In addition, each QiF charging tour in the solution is guaranteed to be valid, *i.e.*, a QiF can return to the ES with non-negative residual energy after charging all the sensors covered by the tour. In addition, each QiF charging tour is executed periodically at certain frequency so that no sensor ever exhausts its battery and the WSN remains operational indefinitely. Such periodic visit and charging is different from [42] and [58], which do not allow a mobile charger to replenish its own battery and thus could not achieve the perpetual operations of a WSN.

It is worth pointing out that JTDP differs from the well-known Vehicle Routing Problem (VRP) and the variants [79]. First of all, the objective of VRP is to find a set of tours with minimum total length. However, in JTDP, we have to consider not only the length of QiF charging tours but also how often QiFs should travel and charge sensors along these tours. Second, in VRP, the demands of customers are assumed to be known *a priori*. In contrast, in JTDP, the demands (*i.e.*, the energy consumption rates) of sensors are determined by how data are routed in the network, which in turn affects tour planning. The coupling of tour planning and data routing makes JTDP harder than VRP.

4.3 Problem Description

In this section, we present a detailed description of JTDP. We first introduce the network model in Section 4.3.1 and then discuss in details how the QiFs charge the sensors in a WSN in Section 4.3.2. Next, we demonstrate the reason that we use the metric of *distance frequency product* to measure the efficiency of QiFs' charging activities in Section 4.3.3. Finally, we discuss why the issue of charging tour planning and the issue of data routing are tightly coupled in Section 4.3.4.

4.3.1 Network Model

We consider a wireless sensor network that is randomly deployed in a two-dimensional terrain. A sensor i periodically senses the environment and generates sensory data at rate r_i. Once generated, sensory data is relayed towards a base station via multi-hop wireless transmissions.

Wireless communication is the major contributor to sensor's energy consumption. In this research, we assume that a sensor's energy consumed for overhearing MAC-layer PDUs not destined to the sensor can be ignored, since such energy consumption can be effectively reduced by MAC schemes such as S-MAC [89]. Energy consumed for sensing and data processing is also ignored. Consequently, the energy consumption rates of sensors are determined by data forwarding, *i.e.*, the rates at which sensors transmit and receive data.

4.3.2 QiF Charging Tour

A QiF initially resides at an ES. Periodically, a QiF departs from the ES with a full battery, moves from one sensor node to another, charges all visited sensors, and returns to the ES. Upon returning to the ES, the battery of a QiF is recharged or replaced. The trajectory of a QiF's movement is termed *QiF charging tour*. In other words, a QiF charging tour is a cycle that connects the ES with a subset of sensors.

As the operations of a QiF are battery powered, we need to ensure that the energy consumed for executing a charging tour, including both the energy for QiF's movement and the energy for wirelessly charging sensors, does not exceed QiF's battery capacity. If a QiF

charging tour satisfies the this condition, the tour is said to be *valid*. Specifically, a valid charging tour is a tour that the QiF could return to the ES with non-negative residual energy after charging an appropriate number of sensors along the tour.

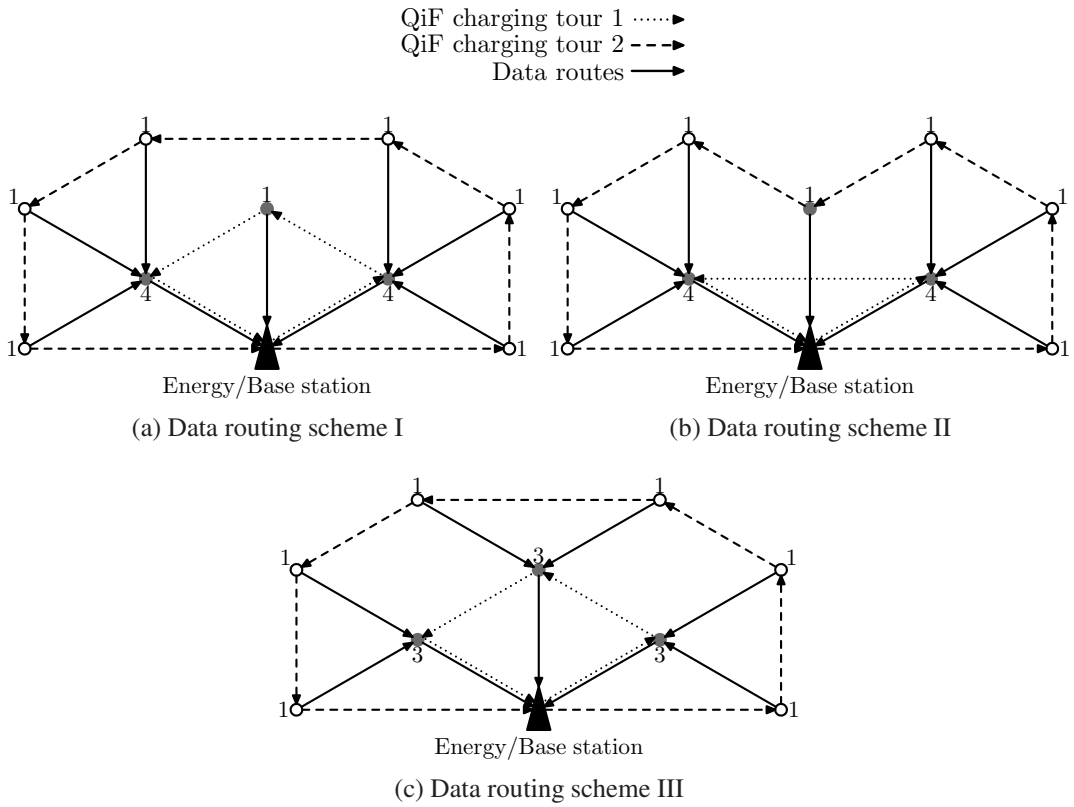
To enable sensors to operate for an unlimited time, QiF charging tours have to be periodically executed to replenish sensor batteries. The *execution frequency* of a QiF charging tour is determined by the energy consumption rates and the battery capacities of the sensors covered by the tour. Given the energy consumption rate e_i and the battery capacity B_i, sensor i must be charged at the frequency of $\frac{e_i}{B_i}$ (termed *charging frequency* of sensor i). In this research, we assume all sensors in a WSN are equipped with batteries of the same capacity B_s, so that the charging frequency of each sensor is solely determined by the sensor's energy consumption rate. Therefore, the execution frequency of a QiF charging tour must be no less than the maximum charging frequency of the sensors covered by that tour. Otherwise some sensors would die between consecutive visits of a QiF.

4.3.3 Distance Frequency Product

Contrasting to the energy that a QiF consumes on wireless charging, the energy consumed on travel is obviously a waste, as such energy is not utilized to charge sensor batteries. Consequently, it is inefficient when QiFs consume unnecessary energy on travel. To characterize the QiFs' energy consumption on travel, let us assume a WSN starts operating at time 0, and all sensors in the WSN are covered by a set Γ of QiF charging tours. Each charging tour $\tau \in \Gamma$ has an execution frequency $F(\tau)$ and a traveling distance $L(\tau)$. Up to time t, the total amount of energy consumed by QiF(s) on travel to execute charging tours in Γ can be expressed as

$$C_t \sum_{\tau \in \Gamma} L(\tau) F(\tau) t,$$

where C_t is the energy consumed by a QiF to travel one unit distance and $F(\tau)t$ denotes how many times tour τ has been executed over the time interval $[0, t]$. We term the coefficient $\sum_{\tau \in \Gamma} L(\tau) F(\tau)$ the *Distance Frequency Product* (DFP) and use DFP to measure the quality of the QiF charging tours. It is easy to see that DFP considers both the traveling distance of



(a) Data routing scheme I

(b) Data routing scheme II

(c) Data routing scheme III

Figure 4.1: The coupling of QiF charging tours planning and sensory data routing

the QiF charging tours and how often these tours are executed. A high DFP value indicates that the QiFs consume (and hence waste) more energy on travel. In our research, we try to design the QiF charging tours with low DFP value so as to reduce QiFs' energy consumption on travel and thus boosting the QiFs' efficiency.

To reduce energy consumption on travel, it is desirable to reduce the frequency that a sensor is visited (and charged). Ideally, a QiF should charge a sensor right before the sensor runs out of energy. Also, it is advantageous to plan a charging tour that covers those sensors consuming energy at similar rates so that, when a QiF starts executing a charging tour, all of these sensors (to be) visited on the charging tour are low in their energy levels and no sensor will be visited and charged *prematurely* (*e.g.*, when a sensor's battery level is more than 30%).

Fig. 4.1 depicts a simple WSN of nine sensors with the same sensory data generation rate of one unit of sensory data per second. The six white sensors use the three gray sensors

to relay data to the ES. By ignoring the energy consumption of data reception, each gray sensor both relays data from two upstream sensors and transmits the data generated by the gray sensor itself so that the energy consumption rate of a gray sensor would be three times that of a white sensor. When a single charging tour is planned to cover all of the sensors, the white sensors are unnecessarily visited as frequently as the gray sensors. That is, the white sensors are visited when they still have plenty of energy left, so that the energy consumed by a QiF to travel to the white sensors is wasted. The single tour solution of [69] suffers from this exact problem, where the need to execute the charging tour is dominated by few *most vulnerable* (or bottleneck) sensors. Each time a vulnerable sensor is about to exhaust its battery, other sensors will be *unnecessarily* visited and charged when their residual energy levels are still high. We argue that a better solution would be to plan two shorter QiF charging tours (as tour 1 and tour 2 in Fig. 4.1(c)), so that the white sensors are charged less frequently (*i.e.*, tour 2 is executed less frequently) than the gray sensors (tour 1).

4.3.4 The Coupling of Tour Planning and Data Routing

Given the intuition that a QiF charging tour should cover sensors draining energy at a similar rate, the tight coupling of QiF charging tour planning and routing of sensory data may best be illustrated by contrasting Figs. 4.1(a), 4.1(b), and 4.1(c) of the same topology but with different routing schemes. In Fig. 4.1, nine sensors and the ES are placed on an equilateral triangle grid with side length of 10 meters. Two charging tours are planned with each routing scheme. For routing schemes I and III, tour 1 (of distance 40 m) covers all of the four gray nodes (including the ES), and tour 2 (of distance 92 m) covers all of the six white nodes. In contrast, for routing scheme II, tour 1 (of distance 37 m) covers two gray nodes (including the ES), and tour 2 (of distance 95 m) covers all of the six white nodes and one gray node. For illustration, we assume that battery capacity B_s is 300 J, sensors consume 1 J of energy to transmit one unit of data, and all sensors generate sensory data at the same rate of one unit of sensory data per second. The total energy consumption rate of each sensor (to transmit both locally generated and relayed sensory data) is labeled beside the sensor in the figure.

Notice that even with different routing schemes, the total energy consumption rate of all of the sensors in the network remains the same ($\sum_i e_i = 15$ J/s). That means the QiF wirelessly charges the network with the same total amount of energy no matter which routing scheme is adopted. However, DFP for Fig. 4.1(a) is 0.84 m/s ($40 \times \frac{4}{300} + 92 \times \frac{1}{300}$), DFP for Fig. 4.1(b) is 0.81 m/s ($37 \times \frac{4}{300} + 95 \times \frac{1}{300}$), and that for Fig. 4.1(c) is 0.71 m/s ($40 \times \frac{3}{300} + 92 \times \frac{1}{300}$). Clearly, routing scheme III results in less energy being consumed by the movement of QiF, and is preferred.

The example demonstrates the tight coupling of data routing and charging tour planning so that both should be jointly optimized. However, the difficulty of JTDP stems from the fact that both the optimal number of QiF charging tours to be planned and which sensors should be covered by which tours are unknown. Moreover, the quality of the designed QiF charging tours is measured by the DFP, which considers both the length and the execution frequency of tours. As the execution frequency of tours is determined by the energy consumption rate of the sensors covered by the tour, it is necessary to carefully determine how data is routed by the sensors towards the ES.

4.4 Problem Formulation

We formulate JTDP in a Mixed Integer Linear Programming (MILP) model. A WSN is modeled as a complete graph with a set of vertices $\aleph = \{0, \cdots, n\}$, where vertex 0 denotes the ES and all other vertices represent the sensors. The model computes an optimal set Γ of QiF charging tours where each sensor is covered by exactly one tour. Each tour $\tau \in \Gamma$ covers an ordered subset of sensors $C(\tau) = \{s_1, \cdots, s_k\}, 1 \leq k \leq n$, such that the sequence of edges $0s_1, s_1s_2, \cdots, s_k0$ form tour τ. The frequency of tour τ, $F(\tau)$, is determined by the charging frequency of the sensors on tour τ. Given sensor i's energy consumption rate e_i and the battery capacity B_s, the charging frequency RC_i of sensor i is $\frac{e_i}{B_s}$. Therefore, the frequency of a tour must be no less than the charging frequency of any sensor covered by that tour to ensure that no sensor dies out of energy before the next visit of the QiFs:

$$F(\tau) \geq \max_{i \in C(\tau)} RC_i. \tag{4.1}$$

To facilitate the MILP formulation, we associate a frequency with each edge between two vertices that can be potentially traveled over by a QiF. If an edge is included in a QiF charging tour, the frequency of the edge is equal to the frequency of the tour. Let f_{ij} denote the frequency of the edge from sensor i to sensor j when the edge is traveled by tour τ. We have $F(\tau) = f_{0s_1} = f_{s_1s_2} = \cdots = f_{s_k0}$, where $C(\tau) = \{s_1, s_2, \cdots, s_k\}$. We use a binary variable x_{ij} to denote whether the edge from sensor i to sensor j is included in a QiF charging tour. The frequency of an edge would be zero if the edge is not included in any tour, *i.e.*, $f_{ij} = 0$, if $x_{ij} = 0, \forall i, j \in \aleph, i \neq j$. The objective of the MILP model is to find an optimal set of QiF charging tours to minimize DFP. The DFP metric over a set of QiF charging tours Γ is defined as follows.

$$DFP(\Gamma) = \sum_{\tau \in \Gamma} L(\tau)F(\tau) = \sum_{i,j \in \aleph}^{i \neq j} d_{ij} f_{ij}, \tag{4.2}$$

where $L(\tau)$ denotes the length of tour τ and d_{ij} is the physical distance between vertices i and j.

We now consider the constraints that ensure each QiF charging tour to be valid. Denoting m_i as the residual energy level of a QiF when departing from sensor i and assuming the QiF visits sensor j immediately after visiting sensor i (*i.e.*, $x_{ij} = 1$), we have

$$m_j - m_i = -\left(C_t d_{ij} + \frac{1}{\eta}\frac{e_j}{f_{ij}}\right), \quad \text{if } x_{ij} = 1, \tag{4.3}$$

where C_t is the energy consumption for a QiF to travel one unit distance and η denotes the wireless energy transfer efficiency. Thus, $C_t d_{ij}$ is the energy consumed by the QiF to travel from sensor i to sensor j, $\frac{e_j}{f_{ij}}$ is the energy consumption of sensor j between two consecutive QiF visits, and $\frac{1}{\eta}\frac{e_j}{f_{ij}}$ is the energy that a QiF needs to charge sensor j every time sensor j is visited by a QiF. Noting that (4.3) is *not linear* as the denominator f_{ij} is a variable, we multiply both sides of (4.3) by f_{ij}. By denoting $m_j f_{ij}$ as t_{ij} and $m_i f_{ij}$ as $\hat{t}_{ij}$, (4.3) can be

B_s	battery capacity of a sensor
B_{QiF}	battery capacity of a QiF
C_t	energy consumption for a QiF to travel one unit distance
d_{ij}	distance between sensor i and sensor j
E_{rx}	energy consumption for receiving one data bit
E_{tx}	energy consumption for transmitting one data bit
e_i	energy consumption rate of sensor i
f_{ij}	frequency that a QiF travels from sensor i to sensor j
G_i	rate of sensory data generated by sensor i
L_c	set of feasible communication links
m_i	residual energy of a QiF upon leaving sensor i
R_i	rate of sensory data received by sensor i
r_{ij}	rate of data transmitted from sensor i to sensor j
T_i	rate of sensory data transmitted by sensor i
t_{ij}	defined as $m_j f_{ij}$
$\hat{t}_{ij}$	defined as $m_i f_{ij}$
u_i	order of sensor i on the tour that covers this sensor
x_{ij}	indicates whether a QiF uses the path from sensor i to sensor j
η	wireless energy transfer efficiency

Table 4.1: Definitions of constants and variables

rewritten as

$$t_{ij} - \hat{t}_{ij} = -\left(C_t d_{ij} f_{ij} + \frac{e_i}{\eta}\right), \quad \text{if } x_{ij} = 1. \tag{4.4}$$

Since $m_0 = B_{QiF}$ as we assume a QiF's battery is fully charged when the QiF departs from the ES, we have $\hat{t}_{0i} = m_0 f_{0i} = B_{QiF} f_{0i}, \forall i \in \aleph - \{0\}$.

Assume sensor s_k is the last sensor visited on the corresponding QiF charging tour. To guarantee each charging tour to be valid, we need to make sure that a QiF has enough energy to travel back to the ES after charging all the sensors covered by the tour. Consequently, we have: $m_{s_k} - C_t d_{s_k 0} \geq 0$, which is equivalent to $\hat{t}_{s_k 0} \geq C_t d_{s_k 0} f_{s_k 0}$.

JTDP is formulated in MILP as follows, which computes an optimal set of QiF charging tours with minimum DFP defined in (4.2). The notations used in the formulation are summarized in Table 4.1.

$$\min \sum_{i,j\in\aleph}^{i\neq j} d_{ij} f_{ij}, \tag{4.5}$$

$$\text{s.t.} \sum_{i\in\aleph-\{0\}} x_{0i} = \sum_{i\in\aleph-\{0\}} x_{i0}, \tag{4.6}$$

$$\sum_{i\in\aleph}^{i\neq j} x_{ij} = \sum_{i\in\aleph}^{i\neq j} x_{ji} = 1, \qquad \forall j \in \aleph - \{0\}, \tag{4.7}$$

$$u_i - u_j + p x_{ij} \leq p - 1, \qquad \forall i, j \in \aleph, i \neq j, \tag{4.8}$$

$$f_{ij} = 0, \qquad \text{if } x_{ij} = 0, \forall i, j \in \aleph, i \neq j, \tag{4.9}$$

$$f_{ij} \geq RC_j, \qquad \text{if } x_{ij} = 1, \forall i \neq j \in \aleph, j \neq 0, \tag{4.10}$$

$$f_{ij} \geq RC_i, \qquad \text{if } x_{ij} = 1, \forall i \neq j \in \aleph, i \neq 0, \tag{4.11}$$

$$f_{ij} = f_{jk}, \qquad \text{if } x_{ij} = x_{jk} = 1, \forall i, j, k \in \aleph, i \neq j, j \neq k, j \neq 0, \tag{4.12}$$

$$\hat{t}_{ij} - t_{ij} = C_t d_{ij} f_{ij} + \frac{e_j}{\eta}, \qquad \text{if } x_{ij} = 1, \forall i \neq j \in \aleph, j \neq 0, \tag{4.13}$$

$$t_{ij} = \hat{t}_{jk}, \qquad \text{if } x_{ij} = x_{jk} = 1, \forall i, j, k \in \aleph, i \neq j, j \neq k, j \neq 0, \tag{4.14}$$

$$\hat{t}_{i0} \geq C_t d_{i0} f_{i0}, \qquad \text{if } x_{i0} = 1, \forall i \in \aleph - \{0\}, \tag{4.15}$$

$$\hat{t}_{0i} = B_{QiF} f_{0i}, \qquad \forall i \in \aleph - \{0\}, \tag{4.16}$$

$$R_i = \sum_{(j,i)\in L_c} r_{ji}, \qquad \forall i \in \aleph - \{0\}, \tag{4.17}$$

$$T_i = \sum_{(i,j)\in L_c} r_{ij}, \qquad \forall i \in \aleph - \{0\}, \tag{4.18}$$

$$R_i + G_i = T_i, \qquad \forall i \in \aleph - \{0\}, \tag{4.19}$$

$$e_i = E_{rx} R_i + E_{tx} T_i, \qquad \forall i \in \aleph - \{0\}. \tag{4.20}$$

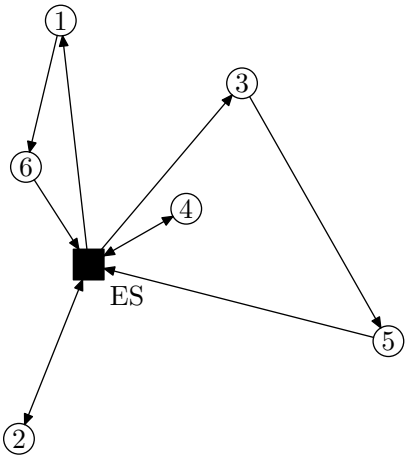
Eq. (4.6) ensures that all tours start from the ES and return to the ES station. Eq. (4.7) guarantees that each sensor is covered by only one tour. Eq. (4.8) serves as the sub-tour elimination constraint that avoids forming degenerated tours that are not connected to the ES. Eq. (4.9) forces the frequency of an unused edge to be 0. Eqs. (4.10) and (4.11) are the

translations of (4.1), which ensure that the frequency that a QiF visits and charges a sensor is higher than or equal to the charging frequency of that sensor. Eq. (4.12) ensures that the traveling edges on the same tour are assigned the same frequency. Eqs. (4.13)-(4.16) guarantee that all tours computed are valid without violating the battery capacity constraint of the QiF. Eq. (4.14) is additionally specified, based on (4.12) and the observation that $t_{ij} = f_{ij}m_j = f_{jk}m_j = \hat{t}_{jk}$. Eqs. (4.17)-(4.20) are constraints that specify data routing. Notice that (4.9)-(4.15) are *conditional constraints* that are not allowed in linear programming languages. The constraints are reformulated using the Big-M method [81] so that the MILP problem can be (theoretically) solved in linear programming kits.

Assuming data routing is known, we have been able to obtain optimal solutions for small networks by solving this MILP problem via the GNU Linear Programming Kit (GLPK). Fig. 4.2 shows a simple network of six sensors, which are one hop away from the ES so that sensory data is transmitted directly from each sensor to the ES. The six sensors generate data at different rates, and the corresponding energy consumption rates are given in Fig. 4.2(b). The optimal solution depicted in Fig. 4.2(a) has four charging tours, which put the sensors with similar energy consumption rates on the same QiF charging tour to minimize DFP. If we simply want to reduce the traveling distance, sensor 4 may be placed on the tour that covers sensors 3 and 5. This practice, however, will cause a QiF to visit sensors 3 and 5 much more frequently than necessary, which increases DFP and makes the solution sub-optimal.

4.5 Heuristic Algorithm

Since an MILP problem is NP-hard, it is impractical to obtain optimal data routing and QiF charging tours for large WSNs by solving the MILP model. Thus, we develop a heuristic algorithm, which first estimates the energy consumption of the sensors in the network and then tries to put the sensors with similar estimated energy consumption rate on the same QiF charging tour. After all tours are designed, data routing is obtained by solving a simple linear programming problem with the objective to minimize DFP.



(a) QiF charging tours

Sensor	Data rate (kbps)	Energy consumption rate (mJ/s)
1	68	141.44
2	40	83.20
3	1	2.08
4	24	49.92
5	2	4.16
6	75	156.00

(b) Data and energy consumption rates

Tour	Distance (m)	Frequency (per hour)
ES→4→ES	40.3	0.1797
ES→2→ES	67.3	0.2995
ES→3→5→ES	255.4	0.0150
ES→1→6→ES	166.9	0.5616

(c) Distance and execution frequency of QiF charging tours

Figure 4.2: An optimal solution for a six-node WSN

We assume that each sensor knows its depth to the ES, *i.e.*, the minimum number of hops from a sensor to the ES. The depth of a sensor can be obtained by growing a minimum spanning tree rooted at the ES over the WSN. We use the depth information to estimate sensors energy consumption rates. Generally speaking, a sensor with lower depth may consume energy at higher rate as the sensor is closer to the ES and is more likely to relay data for other sensors. On the other hand, sensors away from the ES usually have lower energy consumption rates as the sensors only need to transmit data generated by themselves. Under such an observation, the heuristic assumes sensors with the same depth consume energy at the similar rates, and puts these sensors on the same QiF charging tour.

As the sensors covered by each QiF charging tour are expected to consume energy at similar rates, a QiF can visit the sensors when the sensors' batteries are almost empty and fully charge those batteries. Consequently, the energy consumption of a QiF to charge the sensors on a tour could be determined by the traveling distance and the number of sensors visited on the tour. The constraint to guarantee each charging tour to be valid can now be reduced as

$$C_t D + n\frac{B_s}{\eta} \leq B_{QiF},$$

where D is the traveling distance of a tour and n is the number of sensors visited on the tour.

Next the heuristic finds a set of valid QiF charging tours for the sensors of each depth. As such a problem is still NP-hard[4], a simple sub-optimal algorithm is deployed to efficiently design the tours. The algorithm first covers all sensors of a given depth with a set of *unrooted tours* which is an open path allowed to start and end at any two vertices. Then the endpoints of each unrooted tour are connected with the ES/BS to form a complete QiF charging tour. Under the observation that, in dense networks, the sensors of the same depth have similar physically distance to the ES/BS, the energy consumption of an unrooted tour is limited to $B_{QiF} - 2C_t\Delta$, where Δ is the maximum distance of the sensors from the ES.

To illustrate how the unrooted tours are built, consider the set S_k of sensors of depth

[4] Such proof can be conducted by reducing the Distance Constrained Vehicle Routing Problem (DVRP) [38] to the problem of finding QiF charging tours with $B_s = 0$.

k. The algorithm first creates an unrooted tour for each sensor in S_k and checks the edges among the sensors. If an edge can connect two unrooted tours without violating the QiF battery capacity constraint, the edge is inserted into an edge list EL. EL is further sorted into non-decreasing order according the weight of each edge, where the weight is the energy consumption for a QiF to travel along the edge. Then the algorithm iteratively merges two unrooted tours into one. First, the edge of minimum weight is selected from EL. At this moment, the starting point and the ending point of the selected edge belong to different unrooted tours. The two unrooted tours are merged into a single unrooted tour. After the merge, EL is updated such that only the edges that connect two endpoints of different unrooted tours without violating the energy constraints remain in EL. The above process terminates when EL becomes empty, at which point a set of unrooted tours is found. Each unrooted tour is then connected to the depot to form a complete QiF charging tour. To further optimize each tour, the Concorde TSP solver [10] is used to get the best order of sensors to be visited. Since the traveling distance of the optimized tour is no longer than the original tour, the energy constraint is not violated.

Running the above algorithm for each depth k provides a set Γ of charging tours, including the length of each charging tour and the sequence of sensors on each tour. Knowing the set of charging tours, the heuristic still needs to decide the execution frequency of these tours. The frequency $F(\tau)$ of a tour $\tau \in \Gamma$ depends on the energy consumption rates of sensors covered by tour τ, which is determined by the data routing scheme, *i.e.*, the rate at which a sensor transmits and receives data. Thus the heuristic sets up a linear programming problem (LPP) to find a data routing solution that results in the minimum DFP.

$$\min \quad \sum_{\tau \in \Gamma} F(\tau) L(\tau) \tag{4.21}$$

$$\text{s.t.} \quad \sum_{i \in \aleph - \{0\}}^{j \neq i} r_{ij} + G_j = \sum_{i \in \aleph}^{j \neq i} r_{ji}, \qquad \forall j \in \aleph - \{0\}, \tag{4.22}$$

$$e_j = E_{rx} \sum_{i \in \aleph - \{0\}}^{j \neq i} r_{ij} + E_{tx} \sum_{i \in \aleph}^{j \neq i} r_{ji}, \qquad \forall j \in \aleph - \{0\}, \tag{4.23}$$

$$F(\tau) = \max_{j \in C(\tau)} \frac{e_j}{B_s}, \qquad \forall \tau \in \Gamma. \tag{4.24}$$

The inputs to LPP are the set of QiF charging tours Γ, the traveling distance $L(\tau)$ of each tour $\tau \in \Gamma$, and the rate at which each sensor generates data. Eq. (4.22) serves as the flow conservation constraint such that the data transmitted by each sensor equals the data generated and received by the sensor. Eq. (4.23) gives the energy consumption rate of each sensor, and (4.24) guarantees that the frequency of a charging tour is no less than the energy consumption rate of any sensor on the tour. LPP does not include any integer or binary variables, and can be solved efficiently.

4.6 Performance Evaluation

In this section, we evaluate the performance of the heuristic solution. Some interesting properties of the solution are also discussed. We compare our multi-charging tour solution to the single-charging tour solution proposed in [69]. The single-tour solution computes one single charging tour with minimum length, using the Concorde TSP solver, to cover all the sensors in the network. Then a near-optimal solution for both data routing and total cycle time is obtained by solving a formulated MILP problem.

4.6.1 Evaluation Setup

We consider a WSN deployed over a terrain of 500m$\times$500m. The energy station and the base station are co-located at the centroid of the network. A number of sensors are uniformly deployed at random and the sensory data rate of each sensor is randomly and

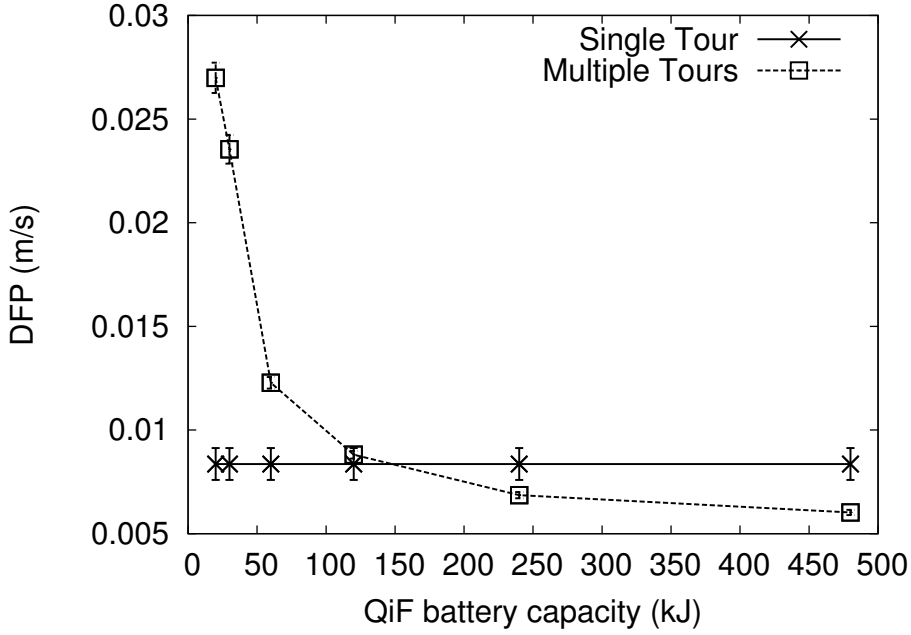


Figure 4.3: DFP vs. battery capacity of QiF

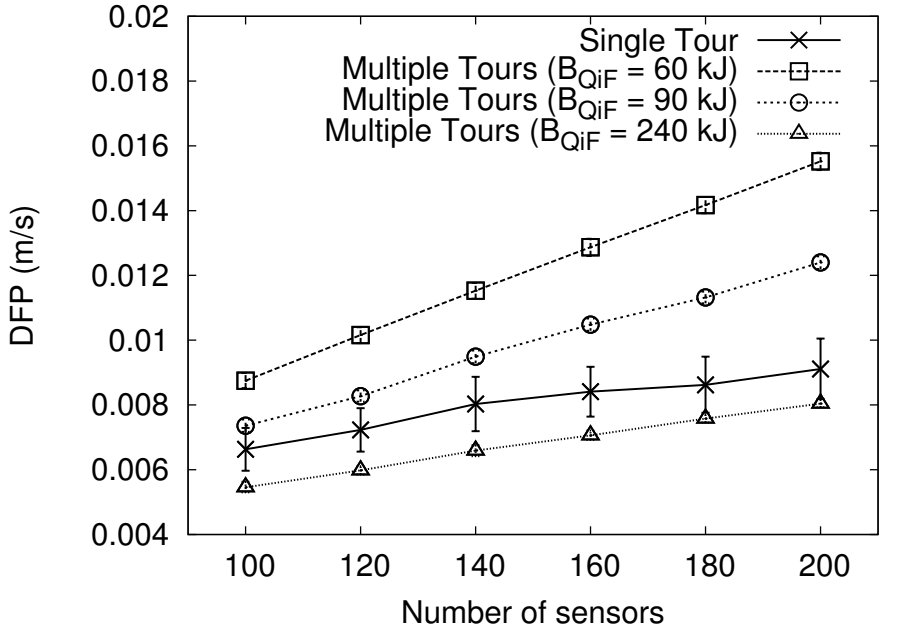


Figure 4.4: DFP vs. number of sensors in the network with different battery capacity of QiF

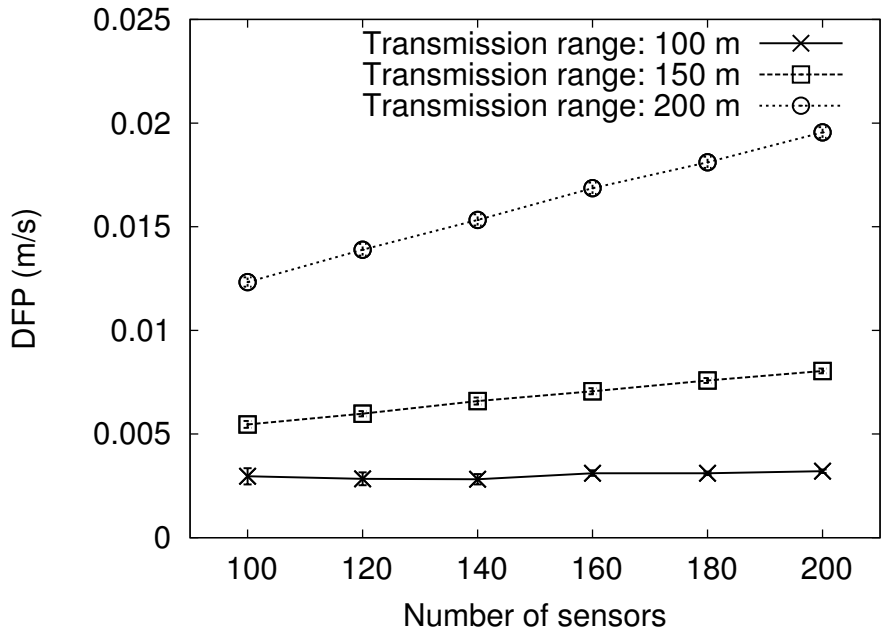


Figure 4.5: DFP vs. number of sensors in the network with different transmission ranges

uniformly selected within (0 kpbs, 10 kbps]. We assume each sensor carries a NiMH battery with capacity B_s of 10.8 kJ. Within a feasible operation range, the efficiency of wireless energy transfer η is assumed to be 80% [35]. The energy consumption for a QiF to move one unit distance (C_t) is 7.69 J/m [51]. All the results presented in Figs. 4.3-4.5 are average of 50 random topologies, and the error bars denote 95% confidence intervals.

In Figs. 4.3 and 4.4, the transmission range of each sensor is set to be 150m, while in Fig. 4.5 we consider how different transmission ranges affect the performance of our algorithm. We assume a simple radio energy model where the transmitter and the receiver dissipate energy to run the radio electronics at the rate of $E_{elec} = 50$ nJ/bit. In addition, a transmitter spends extra energy for the power amplifier with the parameter $\epsilon = 0.0013$ pJ/bit/m^4 [23]. Thus, for a given transmission range d, the energy consumption of a sensor to transmit one bit is $E_{tx} = E_{elec} + \epsilon d^{\alpha}$ where α is the path-loss coefficient and is assumed to be 4.

4.6.2 Effects of QiF's Battery Capacity

We evaluate our heuristic solution with different QiF battery capacities in the network with 150 sensors and the results are shown in Fig. 4.3. The single tour solution is a straight line since the battery capacity of a QiF is not considered. In our heuristic solution, DFP decreases as we increase QiF's battery capacity. This result is due to the fact that with small QiF battery (*e.g.*, less than 50 kJ), a QiF can only charge limited number of sensors per charging tour and has to frequently return to the energy station to refresh its own battery. In the extreme case when $B_{QiF} < 2B_s$, each QiF charging tour covers only one sensor. In such scenarios, the total traveling distance is increased, leading to higher DFP. As the QiF's battery capacity increases, more sensors are to be charged along each tour, and DFP decreases. Once the QiF battery capacity exceeds certain threshold, further increasing the battery capacity does not improve DFP as the sensors of the same depth can always be covered by one charging tour.

It is noteworthy that when the QiF battery capacity is large enough, our multi-tour solution performs better than the single-tour solution. In the single-tour solution, a QiF has to refresh all the sensors' batteries whenever the bottleneck sensor depletes its battery while other sensors still have enough energy to support network activities for a long time. On the other hand, our multi-tour solution tries to put the sensors with similar energy consumption rates on the same charging tour and always charge a sensor when the sensor is about to exhaust its battery. Thus, although the traveling distance of our solution is longer than that of the single-tour solution, our solution presents better DFP when the battery capacity of a QiF is large enough.

4.6.3 Effects of Number of Sensors

We also investigate in Fig. 4.4 the effects of deploying different number of sensors in the network. As the number of sensors increases, the total energy consumption of the network increases as more data is generated and transmitted. Also the traveling distance of the QiF charging tours increases as more sensors have to be covered. These two factors cause DFP to increase as more sensors are deployed in the network.

4.6.4 Effects of Sensor's Transmission Range

Fig. 4.5 shows how the sensor's transmission range affects the performance of our solution. The battery capacity of a QiF is set to be 240 kJ. Increasing the transmission range leads to smaller maximum depth of the sensors in the network and fewer QiF charging tours. Consequently, the total traveling distance of the charging tours decreases. Nevertheless, extending transmission range means that each sensor has to dissipate more energy to run the power amplifier and the increase in energy consumption is roughly proportional to d^4. Thus, DFP increases as the charging tours have to be executed more frequently, even though the total traveling distance of a QiF decreases.

4.7 Summary

The lifetime of WSNs is typically constrained by the battery capacity of sensors. To deploy long-lived WSNs, the research proposes to use the wireless energy transfer technology to replenish sensor batteries. Specifically, we propose Qi-Ferries (QiFs) equipped with energy transmitters that periodically rove the network and wirelessly charge sensor batteries. We demonstrate that the charging efficiency of QiFs can be measured by the distance frequency product (DFP), and the issue of charging tour planning and the issue of sensory data routing should be jointly addressed to minimize DFP. We formulate the joint problem of charging tour planning and sensory data routing in a Mixed Integer Linear Programming model, and develop a heuristic solution. Numerable results validate the efficiency of the heuristic solution.

Chapter 5

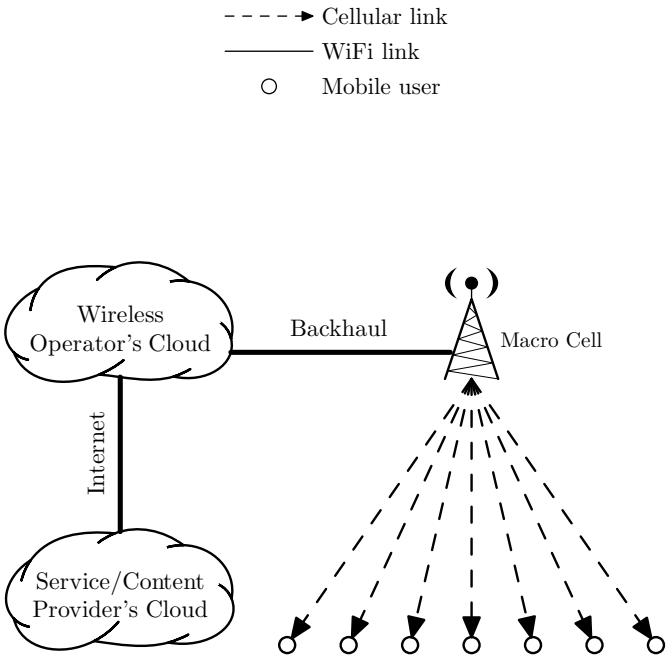
POWER EFFICIENT MOBILE PEER-TO-PEER STREAMING

5.1 Introduction

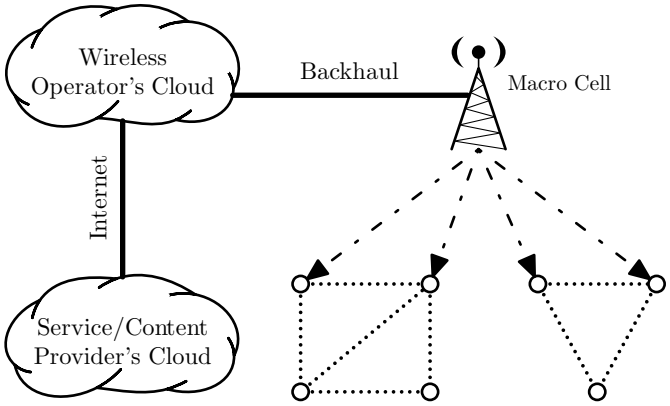
With the proliferation of Internet-capable mobile devices, streaming video to mobile users is becoming more and more popular. Besides the traditional Video on Demand (VoD) services, content providers have developed applications to deliver real-time contents to mobile devices, *e.g.*, WatchESPN [80]. The increasing demand for delivering video traffic to mobile endpoints leads to mobile data explosion. For instance, in 2011, global mobile data traffic grew by a factor of 2.3, more than doubling for the fourth year in a row. Meanwhile, the portion of Internet traffic containing mobile video exceeded 50% for the first time [27]. To confront this challenge, the solutions cannot be simply about allocating more resources at the problem, but rather require radically different approaches to utilizing resources.

In this chapter, we consider delivering live video streaming to a group of co-located mobile users (MUs). For example, a group of football fans in a sport arena may use smartphones to watch instant replays. Also, students in the same residence hall may use tablets to watch an online lecture at the same time. Fig. 5.1(a) depicts such a scenario where the MLB video content is streamed to a group of mobile users (MUs). Originating from a content provider's cloud, the streaming content is transported over the Internet and through the wireless operator's private network to a base station (BS) which then transmits the data to MUs via the cellular links.

While emerging cellular broadband systems, such as 4G LTE, are capable of catering to the bandwidth and latency requirements [39, 70], the Radio Resource Control (RRC) of the cellular systems is not optimized for delivering streaming traffic [8, 30]. Specifically, during a streaming session, the communication component of an MU has almost no chance



(a) Delivering a video content to MUs without P2P streaming



(b) Delivering a video content to MUs with P2P streaming

Figure 5.1: Network infrastructure for delivering video stream to MUs

of entering low-power idle state due to the short inter-arrival time of streaming data PDUs, which leads to high power consumption[1] of the MU. For example, measurement studies [39] have shown that, when using 4G networks to stream video contents to MUs, more than 50% of the system power is consumed by the communication unit.

To overcome this problem, this chapter investigates how the MUs may cooperate with each other so as to optimize MUs' power consumption. In such a cooperative scenario, the co-located MUs form an ad-hoc local wireless network via the MUs' local communication interfaces (*e.g.*, WiFi or Bluetooth). Instead of downloading the entire video stream via cellular links, each MU may only request a *sub-stream* of the video content over the cellular link. The sub-streams are then transmitted and shared with other MUs via the local wireless networks as depicted in Fig. 5.1(b). Such a scenario is termed *peer-to-peer (P2P) streaming to mobile users*.

To decide how cellular and wireless local networks can be jointly utilized, we assume that the information about the networks, such as the topologies and the capacities of communication links, is collected by a BS. Based on the collected information, the BS formulates and solves an optimization problem that minimizes the maximum power consumption of the MUs. The solution is fed back to the MUs and tells each MU how much data to be downloaded from the BS and how much data to be shared with other MUs. We argue the model is practical as wireless operators are adding computation resources to the *edge* of cellular systems (*i.e.*, BSs) to simplify cellular core networks [65].

The remainder of this chapter is organized as follows. Section 5.2 first reviews existing literature related to this research. Then, Section 5.3 gives a high-level description of the proposed P2P streaming scheme and discusses the relative technical issues to be addressed. The P2P streaming problem is formulated into a LP model with the objective of minimizing the maximum power consumption of MUs in Section 5.4, and numerical results are presented in Section 5.5. Finally, Section 5.6 summarizes this chapter.

[1] Throughout the chapter, the term *power* is used to denote the rate at which *energy* is consumed.

5.2 Related Work

This section reviews related work of this research. Specifically, Section 5.2.1 presents the enabling technologies for delivering streaming to mobile users via a P2P paradigm, and Section 5.2.2 discusses why 4G LTE networks are power-inefficient for delivering video traffic.

5.2.1 P2P Streaming to Mobile Users

With the proliferation of Internet-capable mobile devices, streaming video to mobile users is becoming more and more popular. Besides the traditional Video on Demand (VoD) services, content providers have developed applications to deliver real-time content to mobile devices. The examples include WatchESPN [80] and PPTV [59].

While video streaming is usually delivered over UDP-based protocols such as the Real Time Streaming Protocol (RTSP), the HTTP live streaming protocol, which uses TCP, recently gained wide interest as a potential mobile streaming platform [47]. HTTP live streaming can be implemented in P2P fashion as media segments are identified by URLs in a playlist and can be retrieved from different hosts by independent HTTP requests. In [64], the authors presented an implementation of the first P2P HTTP live streaming system.

Other work aims at optimizing the resources to efficiently deliver P2P video traffic over wireless networks. In [75], the authors show that P2P cooperation can be used to provide near-live TV streaming in a resource-constrained mobile environment. In [53], a cross-layer and P2P-based solution is proposed that distributes live video streaming over a mobile ad hoc network by multicasting the streaming content over a spanning tree in an overlay network. In [90], a distributed algorithm, termed LocalTree, constructs a dissemination tree for the static MUs, and at the same time, builds a streaming mesh for the MUs with high dynamics.

5.2.2 Power Management in 4G LTE Networks

4G LTE network offers much higher data rates and much better services than its predecessors. Nevertheless, the power consumption of the MUs has always been a serious issue. To save power in an MU, the Radio Resource Control (RRC) function of the 4G LTE

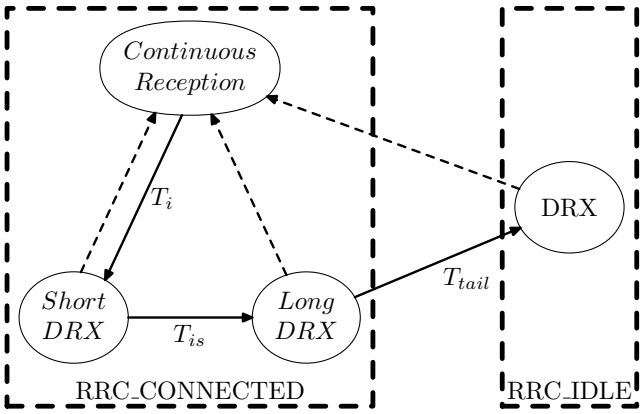


Figure 5.2: RRC state transitions in the 4G LTE network (a dashed arrow denotes a state transition triggered by PDU transmission/reception and a solid arrow represents a state transitions caused by timer expiration)

networks employs Discontinuous Reception (DRX). Fig. 5.2 depicts the transition between two different RRC states (RRC_IDLE and RRC_CONNECTED). Initially an MU is in the RRC_IDLE state. Upon receiving a PDU, the MU is promoted to the RRC_CONNECTED state with a static delay. Specifically, the MU stays in the Continuous Reception mode (of the RRC_CONNECTED state) and starts a DRX inactivity timer (T_i), which is reset by every incoming PDU. When T_i expires, the MU enters into the Short DRX mode and later switches into the Long DRX mode if the MU receives nothing within a certain time period. Finally, the MU may be demoted into the RRC_IDLE state after a long inactive period in the Long DRX mode.

Although aiming to lower the MU's power consumption, DRX is not optimized for streaming traffic with high QoS requirement and low wake-up latency [8, 30]. Although allowed to "micro-sleep" in both the *Short* and the *Long DRX* modes, an MU has to frequently wake up to monitor the Physical Downlink Control Channel (PDCCH), and the power consumption of the MU in these two modes is much higher than that in the RRC_IDLE state. Given that the inter-arrival time of streaming data PDUs is usually short, the MU has little chance of returning to the RRC_IDLE state during the entire streaming session, which leads to high device power consumption. Measurement studies have reported that receiving one unit of data over cellular networks consumes more power than transmitting the same unit of

data over WiFi (or for the same amount of data, transmission over WiFi consumes less power than reception over 4G LTE) [4, 26, 63].

In this chapter, we consider enabling cooperation among co-located MUs using the P2P local wireless networks to deliver live streaming to these MUs in a power-efficient manner. Various studies on using local wireless networks to complement the cellular networks have been conducted. For example, Cellular-WiFi systems use wirelessly inter-connected WiFi access points to extend the coverage of the cellular BSs [21, 60]. There is other work aiming at optimizing the resources to efficiently deliver P2P video traffic over wireless networks. In [75], the authors show that P2P cooperation can be used to provide near-live TV streaming in a resource-constrained mobile environment. In [53], a cross-layer and P2P-based solution is proposed to distribute live video streaming over a mobile ad hoc network by multicasting the streaming content over a spanning tree in an overlay network.

While most existing work uses the P2P networks to reduce MUs' Internet access requirement [29] and to improve the streaming quality [29, 61], less attention has been paid to the power consumption of the MUs. In this research, we take into account the different power efficiencies of the cellular links and the local P2P links, and investigate how streaming content should be shared among MUs so as to reduce their power consumption. Unlike existing work such as [20, 40, 46] where network operators need to supply external rewards (*e.g.*, money) as extrinsic incentives for MUs to facilitate other MUs' communications, we will show that, in our scenario, each MU has intrinsic incentives to cooperate as such cooperation can lower the power consumption and prolong the battery lifetime of the MU.

To the best of our knowledge, [90] is the most related work to this research. In [90], a distributed algorithm named *LocalTree* is proposed to find tree-like routes for static MUs to disseminate video streaming. Meanwhile, LocalTree builds a streaming mesh for the MUs with high dynamics. However, LocalTree only allows one MU (termed as *puller*) in the network to pull data PDUs from the BS, which leads to high power consumption of the puller. In contrast, this research allows multiple MUs to retrieve streaming content from the BS and achieves fair power consumption among cooperating MUs.

5.3 Problem Overview

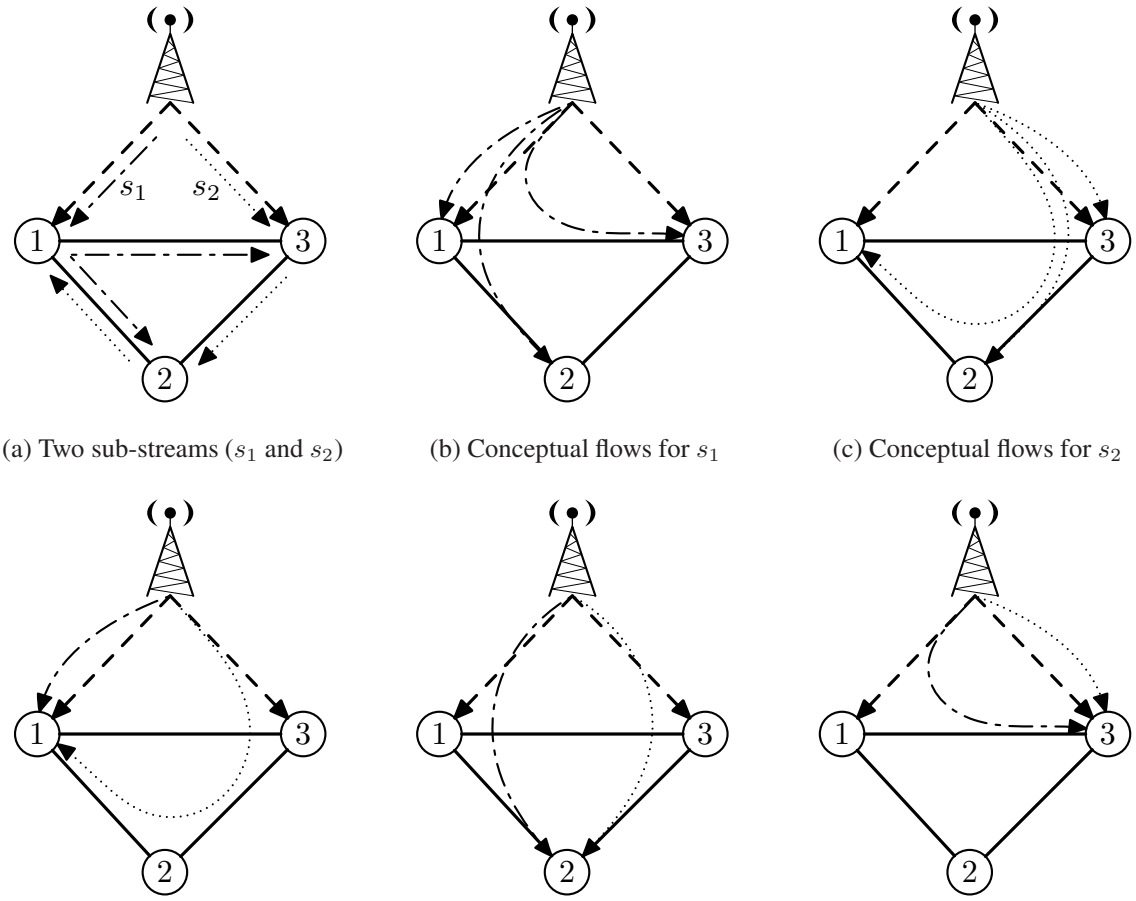
To investigate how cellular and wireless local networks can be jointly utilized for improving power consumption of mobile users, we assume that the topology of MUs is fixed and can be modeled as a directed graph, with the capacity and the power efficiency of the communication links known *a priori*. We envision that such information is collected by the Wireless Operator, where an optimization problem to minimize the maximum power consumption of MUs can be formulated and solved to determine how much streaming data an MU should download over the cellular link and how to share the received streaming data with other MUs via the wireless local networks in the most power efficient manner.

Given that each MU is equipped with multiple radio access technologies, to receive streaming content, an MU can either download the content from a base station or use a wireless local area network (WLAN) such as WiFi[2] to receive the streaming from other neighboring MUs. In such a situation, each MU may only request a portion of the video stream from the base station, and shares the received content with other MUs via WLANs. In the meantime, an MU receives the remainder of the streaming content from its neighboring MUs so that the MU can reconstruct the original content.

Our goal is to investigate P2P video streaming with respect to the power consumption of MUs. Particularly, we aim to determine the P2P cooperative strategy for delivering streaming to MUs that achieves the minimum MUs' power consumption. To facilitate such a usage scenario, the following issues must be jointly addressed:

1) **Streaming data routing**. We use the *fluid flow model* [33] to model video streaming with an encoding rate of S (bits/second). With this model, the original streaming can be divided into "sub-streams" with arbitrary data rates, and each sub-stream may be independently routed over the network following different paths. An MU receives the streaming from multiple sub-streams and combines them together. When receiving streaming content from multiple sub-streams at an aggregated rate of S, an MU can reconstruct the original streaming content. We aim to achieve the *global streaming* of rate S in the network such

[2] In this research, we assume that WLANs are WiFi (IEEE 802.11) networks, but the formulations and solutions can be generalized to other WLAN technologies.



(a) Two sub-streams (s_1 and s_2) (b) Conceptual flows for s_1 (c) Conceptual flows for s_2

(d) Conceptual flows entering node 1 (e) Conceptual flows entering node 2 (f) Conceptual flows entering node 3

Figure 5.3: An example of sub-streams and conceptual flows

that all MUs can playback the streaming content at the original encoding rate. For example, in Fig. 5.3, a stream is divided into two sub-streams and each sub-stream traverses the network over a different path. Global streaming is achieved when $s_1 + s_2 = S$, with s_1 and s_2 denoting the data rates of the two sub-streams, respectively.

To reduce each MU's power consumption, avoiding data duplication is desirable. Ideally each sub-stream needs to be routed over some tree structure imposed on the network so that no MU receives the same content more than once. Consequently, routing the sub-streams is equivalent to assigning a sub-stream data rate to each possible tree in the network. Additionally, we need to ensure that the sum of the data rates of the trees each MU joins is equal to S, so that every MU can reconstruct the original streaming. However, because the number of possible trees is exponential in the size of the network, simply listing all the trees is inefficient. Thus, we also make use of the notion of *conceptual flow* [41] to break a sub-stream tree into multiple unicast conceptual flows from the base station to the MUs, and each MU receives streaming data from multiple conceptual flows. For example, Fig. 5.3(b) shows that the sub-stream s_1 is broken into three unicast conceptual flows and Fig. 5.3(d) shows that two conceptual flows are delivering streaming data to MU 1. Several conceptual flows can share the same physical link without contending for the link capacity, while the actual data rate carried over the physical link is determined by the maximum rate of the conceptual flows over the link. As indicated in [83], it is possible to allocate the streaming content to be delivered over each physical link with the knowledge of the rate of the conceptual flows.

2) **Local P2P link capacity.** When sharing the streaming content with neighboring MUs over WLANs, we need to consider the limited capacity of the P2P links, *i.e.*, each MU has limited capacity in transmitting data over the outgoing P2P links and receiving data over the incoming P2P links. We do not consider the 'broadcast advantage' of wireless transmissions (where one source can send data to multiple destinations using one transmission) since most current P2P applications use application-level multicast, *i.e.*, multicast is realized by multiple concurrent IP unicast sessions managed by the applications. Also, to manage transmission conflicts, we assume there is a perfect transmission scheduling algorithm to avoid transmission contentions in P2P links.

3) **MU power consumption model.** We take into account the MUs' communication power consumption for both the cellular network and the local P2P network. We model the power consumption of an MU by using the power consumption in each RRC state and the percentage of time the MU stays in the state. While the power consumption in each state is usually fixed and determined by the hardware, an MU's sojourn time in each state is determined by the RRC timeout mechanism as well as by the traffic pattern. Thus, accurately modeling an MU's power consumption is difficult and requires knowledge about the average size of PDUs and the inter-PDU delay. In our study, we adopt the simplified model proposed in [26] and assume that the average power consumption P is linearly proportional to the throughputs, *i.e.*,

$$P = \alpha_d^C t_d^C + \alpha_u^W t_u^W + \alpha_d^W t_d^W + \beta, \tag{5.1}$$

where t_d^C, t_u^W and t_d^W are the throughput in the cellular downlink, the P2P outgoing links, and the P2P incoming links, respectively, and α_d^C, α_u^W and α_d^W are the power efficiencies on the corresponding links. β is the base power consumption of an MU. Notice that, in the following discussion, we do not consider the traffic in the cellular uplink, since, in most existing streaming applications, the data traffic mainly flows from the base station to the MUs.

5.4 Problem Formulation

We model the network as a directed graph $\mathcal{G} = (\mathcal{N} + \{0\}, \mathcal{E})$ with $\mathcal{N} = \{1, \cdots, N\}$. Vertex 0 denotes the base station and the vertices in $\mathcal{N}$ denote MUs. Each MU is a receiver of the streaming. The edge set $\mathcal{E}$ models the communication links. A directional edge that originates from vertex 0 is a cellular downlink while the other edges are the local wireless links between MUs. For each communication link $< i, j > \in \mathcal{E}$, variable t_{ij} is introduced to denote the rate of the traffic carried by the link. Also, f_{ij}^k is used to denote the rate of the conceptual flow from the base station to the kth MU over link $< i, j >$.

The optimization problem is formulated as a linear programming problem, shown in

(5.2)-(5.7) as follows.

$$\min_{\{t_{ij}, f_{ij}^k\}} \quad P_{\max} \tag{5.2}$$

$$\text{s.t.} \quad \alpha_d^C t_{0i} + \alpha_u^W \sum_{j \in \mathcal{N}}^{<i,j> \in \mathcal{E}} t_{ij} + \alpha_d^W \sum_{j \in \mathcal{N}}^{<j,i> \in \mathcal{E}} t_{ji} \leq P_{\max}, \qquad \forall i \in \mathcal{N} \tag{5.3}$$

$$\sum_{j \in \mathcal{N}}^{<i,j> \in \mathcal{E}} t_{ij} \leq W^O, \quad \sum_{j \in \mathcal{N}}^{<j,i> \in \mathcal{E}} t_{ji} \leq W^I, \qquad \forall i \in \mathcal{N} \tag{5.4}$$

$$t_{ij} \geq f_{ij}^k, \qquad \forall < i, j > \in \mathcal{E}, \forall k \in \mathcal{N} \tag{5.5}$$

$$\sum_{j \in \mathcal{N}+\{0\}}^{<i,j> \in \mathcal{E}} f_{ij}^k - \sum_{j \in \mathcal{N}+\{0\}}^{<j,i> \in \mathcal{E}} f_{ji}^k = \begin{cases} S & \text{if } i = 0 \\ -S & \text{if } i = k \\ 0 & \text{otherwise} \end{cases}, \quad \forall i \in \mathcal{N} + \{0\}, \forall k \in \mathcal{N} \tag{5.6}$$

$$t_{ij} \geq 0, f_{ij}^k \geq 0, \qquad \forall < i, j > \in \mathcal{E}, \forall k \in \mathcal{N} \tag{5.7}$$

The constraints in (5.3) specify the power consumption of each MU in accordance with the streaming data rates over the cellular link and the P2P links, and $P_{\max}$ denotes the maximum power consumption of all MUs. The constraints limiting the data rates to be carried by the local P2P networks are specified in (5.4). Specifically, for each MU, the aggregated incoming data rate of the P2P links must be less than W^I, and the total outgoing data rates must be less than W^O. Eq. (5.5) ensures that the actual data rate of link $< i, j >$ is the maximum rate of the conceptual flows that use link $< i, j >$. Eq. (5.6) is the flow conservation constraint, *i.e.*, for the kth conceptual flow, the total outgoing flow rate of vertex 0 (the base station) is S and the total incoming flow rate of vertex k is S, while, for the other vertices, the total incoming flow rate is equal to the total outgoing flow rate. Lastly, (5.7) ensures the rates of the conceptual flows over link $< i, j >$ and the physical data rate of link $< i, j >$ are always non-negative.

Our objective is to minimize the maximum power consumption of MUs ($P_{\max}$) subject to the constraints (5.3)-(5.7). We choose to minimize $P_{\max}$ instead of the total power consumption of all MUs because minimizing the total power consumption does not ensure

fairness over the network, *i.e.*, some MUs may have low power consumption while other MUs may consume power at a high level.

5.5 Numerical Results

In this section, we present numerical results to demonstrate that P2P cooperation significantly improves the delivery of video streaming to MUs. We generate different network topologies by uniformly placing a different number of MUs at random in a terrain of size 200m$\times$200m. We assume the local P2P network uses a fixed WiFi communication range of 100 m so that, if two MUs are within the range of each other, they can directly communicate. We only evaluate the network topologies that are connected by WiFi, since each connected component of a partitioned network is considered as a smaller network and evaluated separately. In addition to a WiFi interface, each MU also has a cellular interface to communicate with the BS. In this study, we assume that the quality of a cellular connection does not vary with the location of the MUs and the capacity of each cellular connection is higher than the encoding rate of the streaming traffic. This assumption allows us to simulate the scenario in which each MU downloads the video streaming independently, without cooperation with other MUs.

For each generated network topology, we evaluate the scheme in which MUs cooperate to minimize the maximum per-MU power consumption (P2P:MIN-MAX) and the scheme where there is no cooperation between MUs (NO-P2P). For the first scheme, we set up a linear programming problem as presented in (5.2)-(5.7) and then use the GNU Linear Programming Kit (GLPK) [48] to compute the optimal solution. We also compare our schemes with the LocalTree algorithm [90]. Table 5.1 summarizes the important parameters used in the numerical study. The values of the MU's power efficiencies in different communication links are taken from the measurement study in [63]. Results shown in Figs. 5.4-5.7 are the average of 30 random trials.

Fig. 5.4 shows that, cooperation among the co-located MUs can greatly reduce the maximum MU power consumption can be greatly reduced in comparison to both NO-P2P

Power efficiency in cellular downlink (α_d^C)	40 mW/Mbps
Power efficiency of WiFi transmission (α_u^W)	11 mW/Mbps
Power efficiency of WiFi reception (α_d^W)	5 mW/Mbps
WiFi outgoing link capacity (W^O)	*variable*
WiFi incoming link capacity (W^I)	10 Mbps
Streaming rate (S)	10 Mbps

Table 5.1: Simulation parameters of P2P streaming to mobile users

and LocalTree. Even when there are only two MUs cooperating, a MU's power consumption is saved by about 31% compared to NO-P2P. This reduction in the consumed power is due to the fact that, with cooperation, MUs share streaming content using the more power efficient WiFi networks to reduce the traffic transferred over the cellular connections. Also, notice that the maximum MU power consumption decreases when there are more MUs in the network, which implies that each MU benefits from more cooperation with more neighbors. Although LocalTree also allows cooperation among the MUs, the LocalTree algorithm causes the highest maximum MU power consumption. The reason is that LocalTree only chooses one MU as a *puller* to download all video data PDUs from the BS and then disseminate the PDUs to other MUs. Such a high power consumption may make an MU unwilling to cooperate with other MUs due to the risk of becoming the *puller*.

The LocalTree algorithm outperforms our scheme in terms of the total network power consumption as shown in Fig. 5.5, which is mainly due to the fact that LocalTree allows an MU to *broadcast* a data PDU to multiple neighboring MUs via a single transmission, which reduces total number of data transmissions. We argue that, however, using wireless broadcast may be impractical in realistic scenarios because most current P2P applications use application-level multicast.

Fig. 5.6 illustrates how the outgoing capacity of the WiFi links affects the MU's power consumption. Given that the cellular links are less power efficient, an MU would prefer to cooperate with neighboring MUs to exchange streaming content over WiFi links. However, such cooperation is limited by the capacity of WiFi networks. In other words, when the WiFi outgoing link capacity is small, only limited amount of streaming content can be exchanged locally and each MU has to rely on the cellular network to retrieve additional

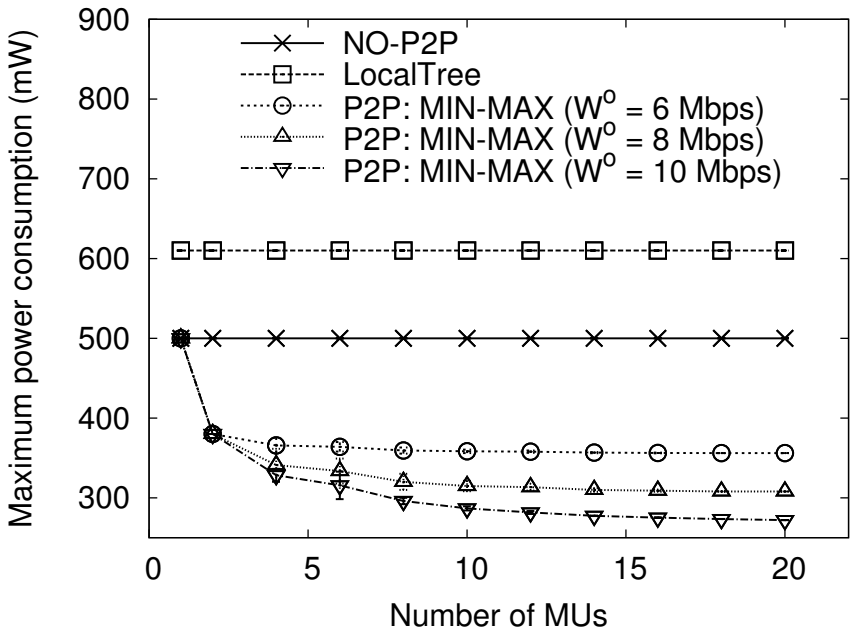


Figure 5.4: Maximum power consumption vs. number of MUs

streaming segments, leading to high cellular network usage (Fig. 5.7) and high power consumption (Fig. 5.6). On the other hand, the power consumption decreases with an increase in the WiFi capacity, as more content can be exchanged locally. The maximum power consumption reaches its minimum when the WiFi outgoing capacity is around 12 Mbps. At this point, the WiFi capacity is no longer the constraining factor. How much data to be downloaded from the BS is determined by the streaming rate S (Fig. 5.7) since a group of MUs only needs to download one copy of the streaming content over the cellular network and can exchange the content locally, resulting in minimum power consumption per MU.

5.6 Summary

In this chapter, we studied the scenario where a group of nearby mobile devices share video streaming content, originating from the cellular network, over wireless local area networks in a P2P manner. To fully facilitate such a usage scenario, we considered streaming data routing, local P2P link capacity, and modeling of MU power consumption. An optimization framework was presented that jointly addresses these three issues and determines

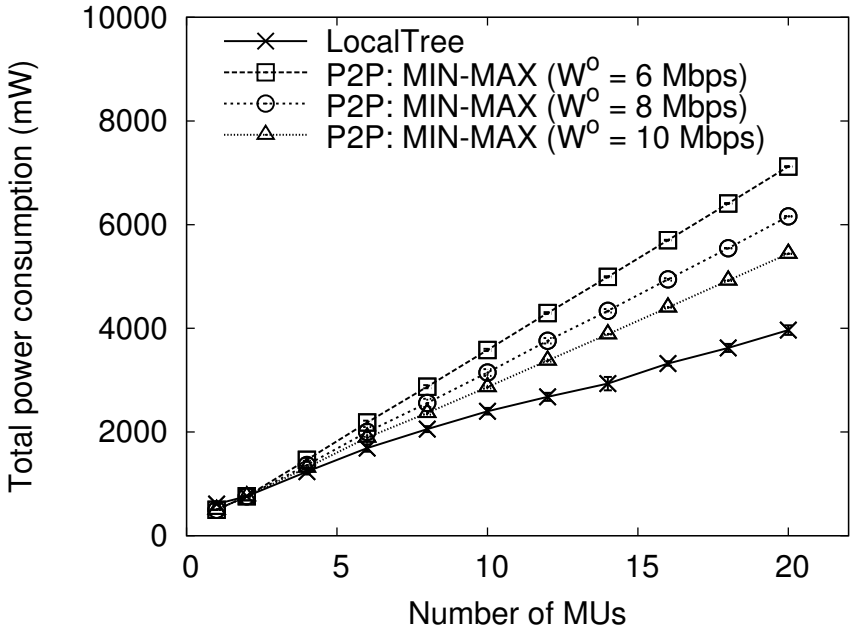


Figure 5.5: Total power consumption vs. number of MUs

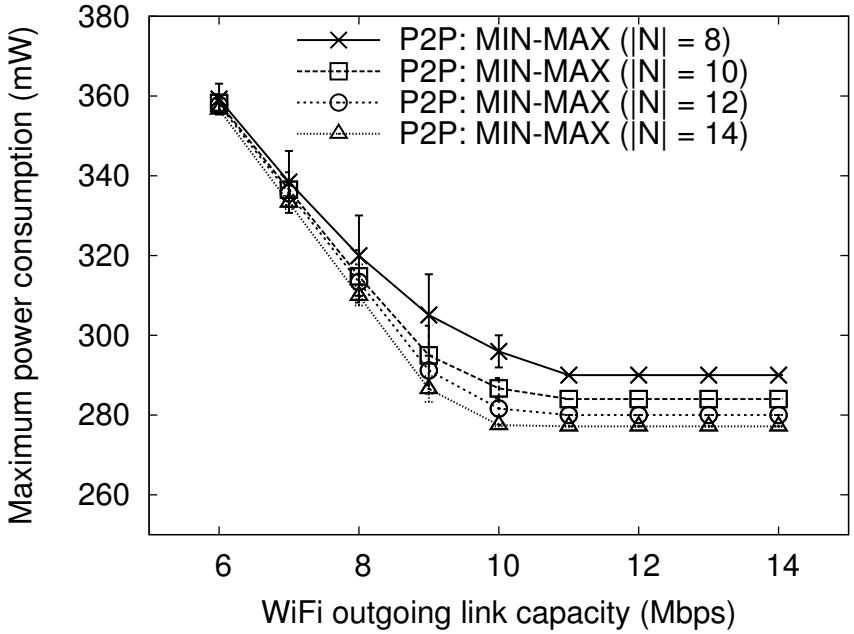


Figure 5.6: Maximum power consumption vs. WiFi outgoing capacity

the optimal P2P cooperation strategy with the objective of minimizing the MUs' power consumption. The numerical results show that a significant amount of power consumption can

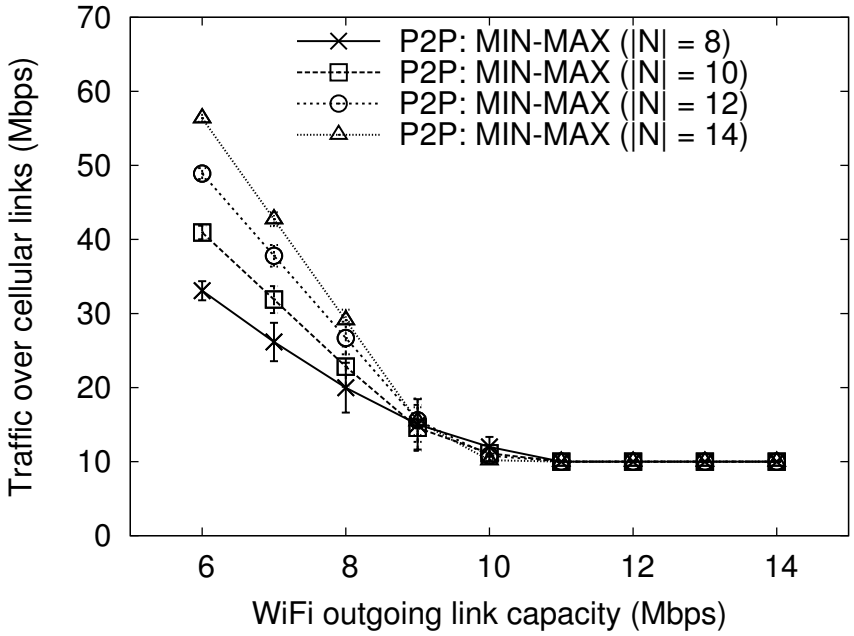


Figure 5.7: Traffic over cellular links vs. WiFi outgoing capacity

be saved as the result of cooperation among mobile devices.

Chapter 6

MOBILITY-AWARE CONTENT CACHING IN SMALL-CELL NETWORKS

6.1 Introduction

We are in the era of a mobile data explosion. Cisco's most recent Visual Networking Index (VNI) [27] reports that global mobile data traffic reached 1.5 exabytes per month at the end of 2013. In addition, mobile data traffic is forecasted to grow at a compound annual growth rate of more than 60% from 2013 to 2018. The ever growing demand from users of mobile wireless networks, in terms of both capacity and coverage, is driving wireless technologies (*e.g.*, LTE, LTE-A, IEEE 802.11ac/ad/af, WiMAX, *etc*.) alike towards their limits. To confront the challenges, one effective approach to boosting both the spatial reuse and the area spectral efficiency is to bring transmitters closer to receivers (*i.e.*, decrease the distance between transmitters and receivers). Motivated by this idea, small-cell networks (SCNs) [24] have been proposed. In SCNs, small-cell base stations (small-BSs), including pico-cell, femto-cell and relay base stations, and WiFi access points, are deployed together with macro-cell base stations as shown in Fig. 6.1. When combined with reusable spectrum and more efficient links, SCNs have the potential to provide the required order-of-magnitude increase in capacity [24].

The emergence of SCNs, however, poses new challenges to the operations of cellular networks. First, unlike macro-cell base stations that are connected to the core network via high-capacity optical fibers, the backhaul bandwidth of small-BSs is usually constrained. For instance, the backhaul of a femto-cell base station is typically facilitated by the customers' home networks such as DSL or cable modem [11]. In addition, due to the reduced cell size and the large number of small cells, mobile users (MUs) may frequently move from one cell to another [45], resulting in rapid fluctuations in the traffic load.

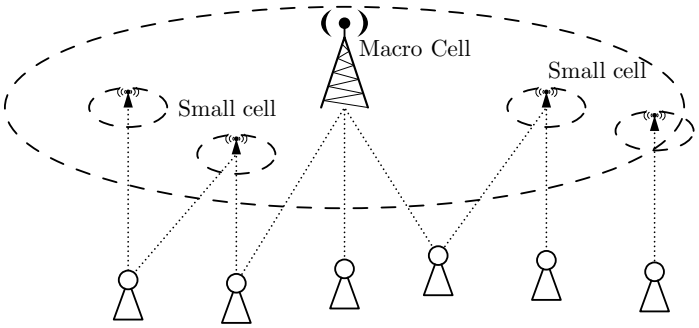


Figure 6.1: A typical small-cell network

The limitations on spectrum bandwidth and per-link spectral efficiency and the challenges of maximizing spatial reuse in SCNs prompt us to investigate solutions beyond the physical layer and network topology. In this chapter, we propose to exploit the application layer semantics of both spatial and temporal locality. Specifically, small-BSs are equipped with a storage facility for caching popular content to tradeoff against constrained backhaul capacity.

The benefits of content caching at small-BSs are two-fold. First, given enough content reuse, most of the requests from MUs can be fulfilled locally, so that much less overhead over the backhaul will be incurred by avoiding retrieving the requested contents from remote servers. Second, caching contents at small-BSs reduces the content retrieval delay. Experiments conducted in [12] have shown that downloading contents from a local cache can be up to 8 times faster than that from a remote server. Such reduction in the retrieval time greatly improves the Quality of Experience (QoE) perceived by MUs, particularly when MUs are highly mobile and have short contact time with small-BSs.

However, given that the storage capacity on a small-BS is finite, only limited contents can be cached. Furthermore, because the *local* preference for particular contents of the MUs currently located within a small cell may be quite different from the *global* popularity of contents, caching decisions made solely based on the global popularity will be sub-optimal. In addition, small cells are densely deployed, and at any moment a single MU may be served by multiple small-BSs. Caching duplicate copies of the same contents at multiple nearby small-BSs not only wastes the cache capacity but also fails to diversify the cached contents

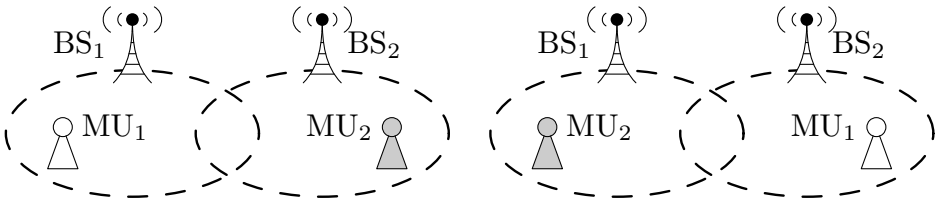
among small-BSs. To address these issues, this chapter studies a mobility-aware content caching problem in SCNs. As to be presented in Section 6.3, we argue that, because MUs may frequently migrate from one small cell to another, small-BSs shall carefully choose which contents to cache by taking the mobility patterns of MUs into consideration, to best serve the MUs currently located within the cell of a small-BS as well as the MUs that may move into the cell in the future.

The chapter proceeds in Section 6.2 to review related work. Section 6.3 motivates the benefits of considering the mobility patterns of MUs when making caching decisions. Detailed network architecture and problem formulation are described in Section 6.4. As the problem is NP-complete, a heuristic solution termed *MobiCacher* with bounded approximation ratio is presented in Section 6.5. We evaluate the performance of *MobiCacher* in Section 6.6, and summarize the chapter in Section 6.7.

6.2 Related Work

To better serve MUs subject to low-capacity backhaul, caching popular contents at small-BSs was first envisioned in [14]. The work considered a group of *stationary* MUs and defined a distributed caching problem that determines which content should be cached by which small-BS and minimizes the average network delay. However, as to be demonstrated in Section 6.3, given that MUs might frequently migrate from one small cell to another and the traffic load of each small cell may fluctuate drastically, a caching solution that ignores MUs' mobility will be sub-optimal.

In [12], a distributed video storage system termed *Sprinkler* was proposed. Motivated by the observation that the contact time between an MU and a small-BS is typically short and the MU may be unable to download one complete video file from a single small-BS, *Sprinkler* breaks a video file into chunks and distributes the chunks to the series of small-BSs along the MU's trajectory. The main objective of *Sprinkler* is to properly cache video chunks so that a particular video chunk can be fetched by the MU before the chunk's playback deadline is due. This work is different from ours as we focus on caching contents to reduce the traffic over backhaul networks.



(a) Time τ (b) Time $\tau + 1$

Figure 6.2: A simple SCN with two small-BSs and two MUs

6.3 Motivating Example

We use an example to motivate the mobility-aware content caching problem. The network pertaining to the example is shown in Fig. 6.2 with two small-BSs (BS_1 and BS_2) and two MUs (MU_1 and MU_2). In time slot τ, MU_1 is located within the cell of BS_1 and MU_2 within BS_2 (Fig. 6.2(a)). In the next time slot $\tau + 1$, both MUs move to the other cell (Fig. 6.2(b)). Both MUs make requests to a library of three objects (O_1, O_2, and O_3). We assume that the MUs' requests are recurrent, *i.e.*, an MU may request the same object at different time slots according to the MU's preference for the object. For instance, users of Spotify [74] who do not turn on the *available offline* option need to download a song each time the song is played. If receiving a request for an uncached object, a small-BS needs to retrieve the object over the backhaul network, which incurs a *cost* (such as the amount of traffic generated in the backhaul network or the amount of energy consumed by the small-BS). Considering the cost c of retrieving an object over a backhaul, and the rate r at which an MU requests the object, we term the product $c \times r$ the *normalized cost* of the object with respect to the MU. The normalized cost of an object with respect to an MU thus represents the expected cost incurred by the MU requesting the object in unit time, if the object is not cached by any small-BS reachable from the MU.

	O_1	O_2	O_3
MU_1	8	1	7
MU_2	1	9	7

Table 6.1: Normalized costs of objects with respect to MU_1 and MU_2

As a part of the example, Table 6.1 lists the normalized costs of the three objects with respect to MU_1 and MU_2. In this example, we assume that each small-BS has the storage capacity to cache one object and an MU's request to a cached object incurs zero cost. In each time slot, the *network cost* is the sum of the normalized costs of the requested but uncached objects. The *total* network cost is the sum of the network costs incurred in time slots τ and $\tau + 1$.

Without considering the mobility of MUs, a small-BS would cache the object with the highest normalized cost with respect to the MU within the cell of the small-BS. In this example, BS_1 would only consider the first row in Table 6.1 and BS_2 the second row. Consequently, BS_1 would choose to cache O_1 and BS_2 O_2. Therefore, the network cost incurred in time slot τ is $(1 + 7) + (1 + 7) = 16$, while the network cost in time slot $\tau + 1$ is $(8 + 7) + (9 + 7) = 31$. The total network cost is 47.

In contrast, if the small-BS considers the mobility of MUs and caches the objects whose normalized costs are high with respect to both the current MUs and future MUs, the total network cost can be reduced. For instance, by considering both rows in Table 6.1 (as each cell will be visited by both MUs in two consecutive slots), both BS_1 and BS_2 will cache O_3 so that the total network cost is reduced to $19 + 19 = 38$. This example clearly shows that taking the mobility of MUs into consideration can improve caching efficiency and reduce the backhaul cost incurred by uncached objects.

6.4 Mobility-Aware Content Caching Problem

We now formally introduce the mobility-aware content caching problem. Given a library of contents, a group of MUs and a set of small-BSs with finite storage capacity, the mobility-aware content caching problem concerns how content should be cached at small-BSs to minimize the total network cost incurred over the backhaul networks. In this section, we first state the assumptions made (Section 6.4.1), and then formulate the problem (Section 6.4.2).

6.4.1 Assumptions

To mathematically formulate the mobility-aware content caching problem, we make the following assumptions.

- MUs' preferences for contents are known *a priori* (for instance, by analyzing MUs' request histories) and supplied as input to the problem. It is also assumed that MUs' preferences for contents remain constant and do not change over a certain period of time. A more realistic model for time-variant user preferences will be considered as future work in Chapter 7.2.3.

- MUs' mobility patterns are also known *a priori*. For instance, MUs' mobility patterns may be specified in their profiles. Also, existing work [52, 56] analyzed MUs' historical information and predicted the future connectivity of the MUs with small-BSs. Our work focuses on utilizing either the specified or the predicted mobility information to facilitate effective content caching at small-BSs. In future, we plan to jointly investigate the content caching problem together with a practical mobility prediction algorithm (Chapter 7.2.1).

- Time is discretized into a sequence of time slots. An MU is assumed to be static within a single time slot, and may move into a different small cell in the next time slot. This assumption allows us to model the continuous movement of an MU using a discrete state in each time slot.

- All of the contents are of an equal size. The cache capacity on a small-BS is specified in terms of the maximum number of contents that can be cached by the small-BS. This assumption leads to a clean formulation of the problem, but can be lifted by introducing auxiliary binary variables that indicate whether heterogeneously sized contents are cached at a small-BS.

- When an uncached content is requested, the content has to be retrieved from a remote server over the backhaul network. When the content arrives at a small-BS, one issue is whether the new content should replace an existing content in the cache. This issue of cache replacement is beyond the scope of this work and will be discussed in Section 7.2

as future work. In this chapter, we assume that, when the cache is full, a newly arrived content will be used once and then discarded, *i.e.*, the content will not replace any existing cached content. This practice is attributed to the fact that more preferred contents will be cached earlier.

6.4.2 Problem Formulation

Consider a small-cell network with a set $\mathcal{F}$ of small-BSs and a set $\mathcal{I}$ of MUs that are interested in a set $\mathcal{L}$ of contents. A small-BS $f \in \mathcal{F}$ is equipped with a storage device that has the capacity to cache up to C_f contents, where C_f is the cache capacity of f. The set of contents cached at small-BS f is denoted as $\mathcal{L}_f$. Obviously $\mathcal{L}_f \subseteq \mathcal{L}$ and $|\mathcal{L}_f| \leq C_f$.

Time is partitioned into a set $\mathcal{T} = \{1, 2, \cdots, |\mathcal{T}|\}$ of consecutive time slots. Given time slot τ, $\mathcal{F}_i^\tau$ describes the set of small-BSs reachable[1] from MU i. Due to mobility, different sets of small-BSs may be reachable from the same MU in different time slots, *i.e.*, $\mathcal{F}_i^{\tau_1}$ may differ from $\mathcal{F}_i^{\tau_2}$ if $\tau_1 \neq \tau_2$.

In any time slot, MU i incurs a normalized cost $c_{i,l}$ if content l is not cached by any small-BS $f \in \mathcal{F}_i^\tau$. As mentioned in Section 6.3, the normalized cost of a content with respect to an MU is the product of the cost of retrieving the content over the backhaul and the rate at which the content is requested by the MU, which thus represents the expected cost of the MU requesting the content in unit time. On the other hand, if in time slot τ, MU i requests a content that is cached by some small-BS in $\mathcal{F}_i^\tau$, zero cost is incurred since the request can be satisfied by a small-BS locally, without incurring any content retrieval over the backhaul. Denoting $\mathcal{O}_i^\tau$ as the set of cached contents that are available to MU i in time slot τ, $\mathcal{O}_i^\tau$ is thus the union of the cached contents at all of the small-BSs reachable from MU i, *i.e.*,

$$\mathcal{O}_i^\tau = \cup_{f \in \mathcal{F}_i^\tau} \mathcal{L}_f.$$

[1] Small-BS f is reachable from MU i if MU i can sense the signal from small-BS f. Given the dense deployment of small-BSs, multiple small-BSs may be reachable from an MU at a given time.

The total network cost is defined as the sum of the normalized costs of the uncached contents incurred by all the MUs over the set of time slots, which is expressed as

$$\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}} c_{i,l}\cdot\mathbf{I}(l\notin\mathcal{O}_\tau^i),$$

where $\mathbf{I}(x)$ is an indicator function that is 1 when x is true, and 0 otherwise. Notice that we can rewrite the total network cost as follows.

$$\begin{aligned}\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}} c_{i,l}\cdot\mathbf{I}(l\notin\mathcal{O}_i^\tau) &= \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}} c_{i,l}\cdot(1-\mathbf{I}(l\in\mathcal{O}_i^\tau))\\ &= \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}} c_{i,l} - \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{O}_i^\tau} c_{i,l}. \qquad (6.1)\end{aligned}$$

The first term of (6.1), $\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}} c_{i,l}$, does not change over different caching decisions, and hence can be regarded as a constant given $\mathcal{T}$, $\mathcal{I}$ and $\mathcal{L}$. Consequently, minimizing the total network cost is equivalent to maximizing the second term in (6.1). We name the second term the *caching utility* $\mathcal{U}$ of a caching scheme. The caching utility in fact represents the cost *saved* by caching contents to avoid retrieving the contents from remote servers over the backhaul.

The mobility-aware content caching problem is then formulated as follows.

$$\max_{\{\mathcal{L}_f\}} \quad \mathcal{U} = \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{O}_i^\tau} c_{i,l}, \qquad (6.2)$$

$$\text{s.t.} \quad \mathcal{L}_f\subseteq\mathcal{L}, \quad |\mathcal{L}_f|\le C_f, \qquad \forall f\in\mathcal{F}, \qquad (6.3)$$

$$\mathcal{O}_i^\tau = \cup_{f\in\mathcal{F}_i^\tau}\mathcal{L}_f, \qquad \forall i\in\mathcal{I}, \quad \forall\tau\in\mathcal{T}. \qquad (6.4)$$

The objective in (6.2) maximizes the caching utility of the caching scheme. The constraint in (6.3) ensures that all cached contents belong to the library $\mathcal{L}$ and the number of cached contents at each small-BS does not exceed the cache capacity C_f. Eq. (6.4) specifies the set of cached contents that are reachable from MU i at time τ.

6.4.3 Computational Intractability

Given the formulation of the mobility-aware content caching problem, we now prove that the problem is NP-complete. To do so, we first cast the optimization problem defined in (6.2)-(6.4) as a decision problem termed the *Caching Decision Problem (CDP)*.

Definition 1 (Caching Decision Problem)**.** *Consider a network with a set $\mathcal{F}$ of small-BSs, a set $\mathcal{I}$ of MUs, a set $\mathcal{L}$ of contents and a set $\mathcal{T}$ of consecutive time slots. A set $\mathcal{F}_i^\tau$ specifies the small-BSs that are reachable from MU i at time τ. An MU i incurs a normalized cost $c_{i,l}$ for content l if l is not cached by any small-BS reachable from MU i. CDP asks whether it is possible to (1) cache contents at small-BSs such that $|\mathcal{L}_f| \leq C_f, \forall f \in \mathcal{F}$, and (2) the caching utility (defined in (6.2)) is larger than or equal to $\mathcal{U}$?*

Obviously, CDP belongs to the class of NP since we can easily calculate the caching utility using (6.2) if the set of contents cached at each small-BS is given as a solution to CDP. To show that CDP is NP-complete, we use a reduction from the k-Disjoint Set Covers (k-DSC) problem, which is known to be NP-complete for any $k \geq 2$ [9, 71]. The definition of the k-DSC problem is as follows.

Definition 2 (k-Disjoint Set Covers)**.** *Given a collection[2] S where each element in S is a subset of a finite set T, determine whether there is a way to partition S into k disjoint sub-collections $\{S_1, \cdots, S_k\}$ such that $S_i \cap S_j = \emptyset, \forall i \neq j$, and each sub-collection S_i covers set T, i.e., $\cup_{s \in S_1} s = \cdots = \cup_{s \in S_k} s = T$.*

The following theorem shows that CDP is NP-complete.

Theorem 6.4.1. *The k-DSC problem is polynomial-time reducible to CDP.*

Proof. Given an instance of the k-DSC problem specified by a set T, a collection S of subsets of T, we construct an instance $\left\{\mathcal{F}', \mathcal{I}', \mathcal{L}', \mathcal{T}', \{\mathcal{F}_i'^\tau\}, \{c'_{i,l}\}, \{C'_f\}, \mathcal{U}'\right\}$ of CDP as follows.

- For each element $t \in T$, create an MU i_t and place i_t in $\mathcal{I}'$, *i.e.*, $\mathcal{I}' \coloneqq \{i_t | t \in T\}$.

[2] Throughout the proof, the term *collection* is used to denote a set/family of sets.

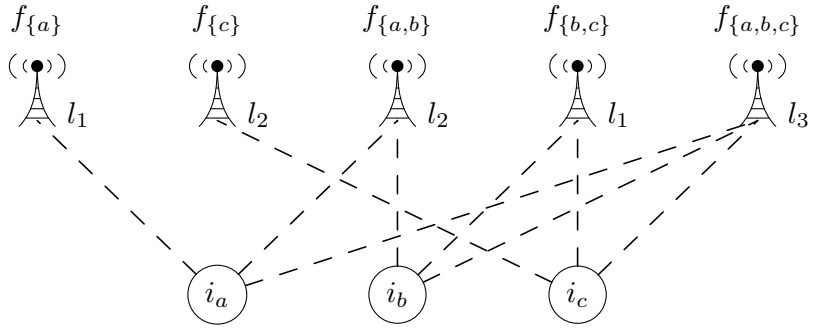


Figure 6.3: An instance of CDP constructed from an instance of the 3-DSC problem. $T = \{a, b, c\}, S = \{\{a\}, \{c\}, \{a, b\}, \{b, c\}, \{a, b, c\}\}$. The content cached by each small-BS is shown besides each small-BS. The corresponding solution to the 3-DSC problem is $S_1 = \{\{a\}, \{b, c\}\}, S_2 = \{\{c\}, \{a, b\}\}$ and $S_3 = \{\{a, b, c\}\}$.

- For each subset $s \in S$, create a small-BS f_s with caching capacity 1, *i.e.*, $\mathcal{F}' \coloneqq \{f_s | s \in S\}$ and $C'_{f_s} \coloneqq 1, \forall f_s \in \mathcal{F}'$.
- Create a library $\mathcal{L}'$ of k contents, *i.e.*, $\mathcal{L}' \coloneqq \{l_1, \cdots, l_k\}$.
- Let $\mathcal{T}'$ contain a single time slot τ, *i.e.*, $\mathcal{T}' \coloneqq \{\tau\}$. Since only one time slot is considered in the constructed instance of CDP, for conciseness, we hereinafter drop any superscript τ in the remaining proof.
- For any element $t \in T$ and any subset $s \in S$, if $t \in s$, then small-BS f_s is reachable from MU i_t. In other words, the set $\mathcal{F}'_{i_t}$ of reachable small-BSs is defined as $\mathcal{F}'_{i_t} \coloneqq \{f_s | t \in s, \forall s \in S\}, \forall i_t \in \mathcal{I}'$.
- The contents' normalized costs with respect to MUs are unanimously set to $\frac{1}{k}$, *i.e.*, $c'_{i_t,l} \coloneqq \frac{1}{k}, \forall i_t \in \mathcal{I}', \forall l \in \mathcal{L}'$.
- Last, set $\mathcal{U}'$ to $|\mathcal{I}'|$, which is the cardinality of $\mathcal{I}$ and equals the cardinality of the set T.

We can perform the above construction process in polynomial time of $O(|C| \cdot |T|)$. We now show that a solution to an instance of the k-DSC problem exists if and only if there is a solution to the constructed instance of CDP.

First, suppose we know a solution to an instance of the k-DSC problem, *i.e.*, a partition of collection S into k disjoint sub-collections $\{S_1, \cdots, S_k\}$. From the solution to the

k-DSC problem, we build a solution to CDP as follows. For each element $s \in S$, if s belongs to sub-collection S_j, we cache content l_j at small-BS f_s. We need to show that with such a caching scheme, the generated caching utility is larger than or equal $|\mathcal{I}'|$. The caching utility can be calculated using (6.2).

$$\mathcal{U} = \sum_{i_t \in \mathcal{I}'} \sum_{l \in \mathcal{O}'_i} c'_{i_t,l} = \sum_{i_t \in \mathcal{I}'} \sum_{l \in \cup_{f_s \in \mathcal{F}'_{i_t}} \mathcal{L}'_{f_s}} \frac{1}{k} = \frac{1}{k} \sum_{i_t \in \mathcal{I}'} \left| \cup_{f_s \in \mathcal{F}'_{i_t}} \mathcal{L}'_{f_s} \right|. \tag{6.5}$$

Since each sub-collection covers set T, we have

$$t \in \cup_{s \in S_j} s, \quad \forall t \in T, \forall j \in [1, k]. \tag{6.6}$$

Thus, for any element $t \in T$ and any sub-collection S_j, there must be a set $s \in S_j$ that contains element t, *i.e.*,

$$t \in s, \quad \exists s \in S_j, \forall t \in T, \forall j \in [1, k]. \tag{6.7}$$

Since $s \in S_j$, small-BS f_s caches content l_j, *i.e.*, $l_j \in \mathcal{L}'_{f_s}$. Due to $t \in s$, we have $f_s \in \mathcal{F}'_{i_t}$. Also, according to the rule we use to construct an instance of CDP from an instance of the k-DSC problem, an element t can be one-to-one mapped to an MU i_t and a sub-collection S_j can be one-to-one mapped to a content l_j. Consequently, we can rewrite (6.7) as

$$l_j \in \mathcal{L}'_{f_s}, \quad \exists f_s \in \mathcal{F}'_{i_t}, \forall i_t \in \mathcal{I}', \forall l_j \in \mathcal{L}'. \tag{6.8}$$

The above expression is equivalent to $\cup_{f_s \in \mathcal{F}'_{i_t}} \mathcal{L}'_{f_s} = \mathcal{L}', \forall i_t \in \mathcal{I}'$ and therefore, $|\cup_{f_s \in \mathcal{F}'_{i_t}} \mathcal{L}'_{f_s}| = |\mathcal{L}'| = k, \forall i_t \in \mathcal{I}'$. Continuing from (6.5), we can now obtain the caching utility as

$$\mathcal{U}' = \frac{1}{k} \sum_{i \in \mathcal{I}'} |\mathcal{L}'| = |\mathcal{I}'|. \tag{6.9}$$

The caching utility $\mathcal{U}'$ is greater than or equal to $|\mathcal{I}'|$. Thus, we have constructed a solution to CDP based on the solution to the k-DSC problem.

Conversely, suppose that there is a caching solution to CDP with a caching utility greater than or equal to $|\mathcal{I}'|$. In fact, the caching utility cannot be larger than $|\mathcal{I}'|$ since

$$\mathcal{U}' = \sum_{i_t \in \mathcal{I}'} \sum_{l \in \mathcal{O}'_{i_t}} c'_{i_t,l} \leq \sum_{i_t \in \mathcal{I}'} \sum_{l \in \mathcal{L}'} c'_{i_t,l} = |\mathcal{I}'|. \tag{6.10}$$

In (6.10), $\mathcal{U}'$ exactly equals $|\mathcal{I}'|$ only if $\mathcal{O}'_{i_t} = \mathcal{L}'$ for any $i_t \in \mathcal{I}'$, *i.e.*, every MU has access to the entire library $\mathcal{L}'$.

Based on the solution to CDP, we construct a solution to the k-DSC problem. Specifically, we partition the collection S into k sub-collections $\{S_1, \cdots, S_k\}$ such that

$$s \in S_j \text{ if small-BS } f_s \text{ caches content } l_j. \tag{6.11}$$

The resulted sub-collections are disjoint, since each small-BS can only cache one content and thus a set s will be placed in exactly one sub-collection. It remains to be shown that each sub-collection covers the set T. Specifically, we need to show that

$$t \in \cup_{s \in S_j} s, \quad \forall t \in T, \forall j \in [1, k]. \tag{6.12}$$

We consider an arbitrary element $t \in T$ and an arbitrary sub-collection S_j with $j \in [1, k]$. Notice that element t can be uniquely mapped to an MU $i_t \in \mathcal{I}'$. Since the maximum caching utility is achieved in the solution to CDP, the set $\mathcal{O}'_{i_t}$ of cached contents available to MU i_t must equal the library $\mathcal{L}'$. Thus the following equation holds.

$$l_j \in \mathcal{O}'_{i_t} = \cup_{f_s \in \mathcal{F}'_{i_t}} \mathcal{L}'_{f_s}, \quad \forall i_t \in \mathcal{I}', \forall l_j \in \mathcal{L}',$$

which is equivalent to

$$l_j \in \mathcal{L}'_{f_s}, \quad \exists f_s \in \mathcal{F}'_{i_t}, \forall i_t \in \mathcal{I}', \forall l_j \in \mathcal{L}'. \tag{6.13}$$

Since $\mathcal{F}'_{i_t}$ is defined as $\{f_s | t \in s, \forall s \in S\}$ and $f_s \in \mathcal{F}'_{i_t}$, we derive that $t \in s, \exists s \in S$. Also,

from $l_j \in \mathcal{L}'_{f_s}$ and (6.11), we know that $s \in S_j$. Thus we can derive from (6.13) that

$$t \in s, \quad \exists s \in S_j, \forall i_t \in \mathcal{I}', \forall l_j \in \mathcal{L}'. \tag{6.14}$$

Since each element $t \in T$ can be uniquely mapped to an MU $i_t \in \mathcal{I}'$ and each content $l_j \in \mathcal{L}'$ can be uniquely mapped to a subset S_j, we can further rewrite (6.14) as

$$\begin{aligned} & t \in s, \quad \exists s \in S_j, \forall t \in T, \forall j \in [1, k]. \\ \Rightarrow \quad & t \in \cap_{s \in S_j} s, \quad \forall t \in T, \forall j \in [1, k]. \end{aligned} \tag{6.15}$$

Eq. (6.15) shows that each sub-collection S_j covers the set T. Consequently, we can conclude the sub-collections $\{S_1, \cdots, S_k\}$ is indeed a solution to the k-DSC problem, which completes the proof of Theorem 6.4.1. □

Combining the fact that CDP belongs to the class of NP and Theorem 6.4.1, we conclude that CDP is NP-complete.

6.5 Heuristic Algorithm

As the mobility-aware content caching problem formulated in (6.2)-(6.4) is NP-complete, in this section, we describe a heuristic solution termed *MobiCacher*. We prove that the worst-case performance of *MobiCacher* is bounded by an approximation ratio F, where F is the maximum number of small-BSs that are reachable from an MU at any time. We also evaluate the performance of *MobiCacher* via trace-based simulation, and present the results in Section 6.6.

The complexity of the mobility-aware content caching problem is largely due to the fact that an MU may access the caches of multiple small-BSs at a time, and thus, deciding whether a content should be cached at one small-BS is affected by what contents are cached by nearby small-BSs. To simplify the decision making, *MobiCacher* decomposes the original problem into $|\mathcal{F}|$ sub-problems, with one sub-problem for each small-BS. Each sub-problem is then solved independently, which enables *MobiCacher* to run in polynomial time.

Let $\mathcal{I}_f^\tau$ denote the set of MUs that may be associated with small-BS f at time slot τ. The sub-problem for small-BS f tries to select a set of contents $\mathcal{L}_f$ that maximizes the caching utility generated by MUs in $\cup_{\tau\in\mathcal{T}}\mathcal{I}_f^\tau$. This sub-problem for small-BS f is formulated as

$$\max_{\mathcal{L}_f} \quad \mathcal{U}_f = \sum_{l\in\mathcal{L}_f}\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}_f^\tau} c_{i,l} \tag{6.16}$$

$$\text{s.t.} \quad \mathcal{L}_f \subseteq \mathcal{L}, \quad |\mathcal{L}_f| \leq C_f \tag{6.17}$$

Notice that $\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}_f^\tau} c_{i,l}$ can be rewritten as $\sum_{i\in\mathcal{I}} T_i^f c_{i,l}$, where T_i^f is the total number of time slots that MU i is associated with small-BS f (termed MU i's *sojourn time* in small-BS f). Thus, to solve the problem defined in (6.16)-(6.17), small-BS f can iteratively select content l with maximal $\sum_{i\in\mathcal{I}} T_i^f c_{i,l}$, until the cache at small-BS f fills up. This process is summarized in Algorithm 3. The complexity of this algorithm is of polynomial time $O(|\mathcal{L}|\cdot|\mathcal{I}| + C_f\cdot|\mathcal{I}|)$. We now analyze the approximation ratio of *MobiCacher* by proving the following theorem.

Theorem 6.5.1. MobiCacher *is a polynomial-time F-approximation algorithm, where F is the maximum number of small-BSs that are reachable from an MU at any time.*

Proof. Let $\mathcal{U}'$ denote the caching utility achieved by *MobiCacher* and $\{\mathcal{L}'_f\}$ the corresponding sets of contents selected to be cached at small-BSs. By the definition of caching utility in (6.2), $\mathcal{U}'$ can be expressed in terms of $\{\mathcal{L}'_f\}$ as

$$\mathcal{U}' = \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{O}'^\tau_i} c_{i,l} = \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\left\{\sum_{l\in\cup_{f\in\mathcal{F}_i^\tau}\mathcal{L}'_f} c_{i,l}\right\}.$$

We introduce the binary variable $B'_{l,f}$ to indicate whether content l is selected by *MobiCacher* to be cached at small-BS f. The membership of each set $\mathcal{L}'_f$ can be fully expressed by the

set $\{B'_{l,f}\}$ of binary variables, and caching utility $\mathcal{U}'$ can be rewritten as

$$\mathcal{U}' = \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\min\left(1,\sum_{f\in\mathcal{F}_i^\tau}B'_{l,f}\right)\right\}. \tag{6.18}$$

The optimal utility $\mathcal{U}^*$ can be expressed similarly by replacing every $B'_{l,f}$ in (6.18) with $B^*_{l,f}$ as follows.

$$\mathcal{U}^* = \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\min\left(1,\sum_{f\in\mathcal{F}_i^\tau}B^*_{l,f}\right)\right\} \tag{6.19}$$

To show that *MobiCacher* is an F-approximation algorithm, we need to show that $\frac{\mathcal{U}^*}{\mathcal{U}'}\leq F$. The remainder of the proof continues from (6.18).

$$\mathcal{U}' \geq \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{O}}\left\{c_{i,l}\cdot\frac{1}{|\mathcal{F}_i^\tau|}\sum_{f\in\mathcal{F}_i^\tau}\min\left(1,B'_{l,f}\right)\right\} \tag{6.20}$$

Eq. (6.20) holds as $\min\left(1,\sum_{i=1}^N a_i\right)\geq\frac{1}{N}\sum_{i=1}^N\min\left(1,a_i\right)$, where each a_i only takes a binary value. We have defined F as the maximum number of small-BSs that an MU can sense at any time, *i.e.*, $F\geq|\mathcal{F}_i^\tau|,\forall i\in\mathcal{I},\forall\tau\in\mathcal{T}$. Thus we have

$$\begin{aligned}\mathcal{U}' &\geq \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\frac{1}{F}\sum_{f\in\mathcal{F}_i^\tau}\min\left(1,B'_{l,f}\right)\right\}\\ &= \frac{1}{F}\sum_{f\in\mathcal{F}}\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\min\left(1,B'_{l,f}\right)\right\}\end{aligned} \tag{6.21}$$

Notice that $\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\min\left(1,B'_{l,f}\right)\right\}$ in (6.21) is the objective function in (6.16) rewritten with the binary variables $\{B'_{l,f}\}$. Since the value of each $B'_{l,f}$ is chosen to maximize (6.16) as opposed to (6.2), the following inequality holds.

$$\textstyle\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\min\left(1,B'_{l,f}\right)\right\}\geq\sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\left\{c_{i,l}\cdot\min\left(1,B^*_{l,f}\right)\right\},\quad\forall f\in\mathcal{F} \tag{6.22}$$

Algorithm 3 *MobiCacher* for small-BS f

Input: A set of candidate contents $\mathcal{L}$; a set of MUs $\mathcal{I}$; MU preferences $\{c_{i,l}\}$; MU sojourn times $\{T_i^f\}$.

1: $\mathcal{L}_f \leftarrow \emptyset$
2: **while** $|\mathcal{L}_f| < C_f$ **do**
3: $\quad l \leftarrow \text{argmax}_l \sum_{i\in\mathcal{I}} T_i^f c_{i,l}$
4: $\quad \mathcal{L}_f \leftarrow \mathcal{L}_f \cup \{l\}$
5: $\quad \mathcal{L} \leftarrow \mathcal{L} \setminus \{l\}$
6: **end while**
7: **return** $\mathcal{L}_f$

Combining (6.21) and (6.22) we have

$$\begin{aligned}
\mathcal{U}' &\geq \frac{1}{F} \sum_{f\in\mathcal{F}} \sum_{\tau\in\mathcal{T}} \sum_{i\in\mathcal{I}} \sum_{l\in\mathcal{L}} \left\{ c_{i,l} \cdot \min\left(1, B_{l,f}^*\right) \right\} \\
&= \frac{1}{F} \sum_{\tau\in\mathcal{T}} \sum_{i\in\mathcal{I}} \sum_{l\in\mathcal{L}} \left\{ c_{i,l} \cdot \sum_{f\in\mathcal{F}_i^\tau} \min\left(1, B_{l,f}^*\right) \right\} \\
&\geq \frac{1}{F} \sum_{\tau\in\mathcal{T}} \sum_{i\in\mathcal{I}} \sum_{l\in\mathcal{L}} \left\{ c_{i,l} \cdot \min\left(1, \sum_{f\in\mathcal{F}_i^\tau} B_{l,f}^*\right) \right\}
\end{aligned} \tag{6.23}$$

The last inequality in (6.23) holds due to the fact that $\sum_{i=1}^N \min(1, a_i) \geq \min\left(1, \sum_{i=1}^N a_i\right)$, where each a_i is a binary variable. Combining (6.23) with (6.19), it is readily seen that $\mathcal{U}' \geq \frac{1}{F}\mathcal{U}^*$, *i.e.*, $\frac{\mathcal{U}^*}{\mathcal{U}'} \leq F$. □

6.6 Performance Evaluation

6.6.1 User Mobility Model

We evaluate the performance of *MobiCacher* by adapting the real trace of mobile users collected by the Wireless Topology Discovery (WTD) project at UCSD [49]. In the WTD trace, an active device records all the access points (APs) that the device could sense across all frequencies at each sampling point. The time interval between two adjacent sampling points is 20 seconds. We thus set the length of a time slot to be 20 seconds in our simulations. We choose to use the WTD trace for our simulation because (1) the dataset is

publicly available, and (2) the WTD trace records multiple APs visible to an MU at each sampling point, while most of the other traces (*e.g.*, [5] and [73]) only record the AP that an MU is associated with at any time. Our use of the WTD trace in the simulation substitutes small-BSs for WiFi APs.

The entire WTD trace includes data from 275 PDA devices accessing more than 400 unique APs during the time interval from 2002/9/22 to 2002/12/08. We choose the busiest day of the trace, namely the day of 2002/10/16, during which the highest number of active MUs was recorded. To evaluate *MobiCacher* under different MU mobility patterns, we run the simulation over four different one-hour time intervals starting at 00:00:00, 06:00:00, 12:00:00 and 18:00:00. Fig. 6.4 plots the cumulative distribution of MUs' sojourn times in small-BSs in these four time intervals. We can see that MUs are more mobile (with shorter sojourn time) during the daytime than that during the night time.

6.6.2 User Preference Model

To model MUs' preferences for contents, we use the song listening logs collected by Last.fm [7]. The Last.fm dataset contains the song listening history of about 1000 unique MUs. We select the 200 top-played songs and put the songs into the library $\mathcal{L}$ of contents. We rank an MU's preference for a song in $\mathcal{L}$ as how many times the MU listened to the song divided by the total number of songs the MU listened to. For example, if an MU listened to a particular song 5 times and the total number of songs the MU listened to is 100, then the MU's preference for this song is 0.05. Moreover, if the MU can retrieve the song from one of the caches of reachable small-BSs in a time slot, the MU gains a utility of 0.05. To evaluate the performance of *MobiCacher*, we randomly associate an MU in the real traces generated in Section 6.6.1 with a listening profile and calculate the MU's preference for songs using the rule mentioned above.

6.6.3 Performance of *MobiCacher*

We compare the performance of three schemes: (1) *MobiCacher* – our mobility-aware caching solution, (2) *FemtoCacher* – the caching solution proposed in [14] where

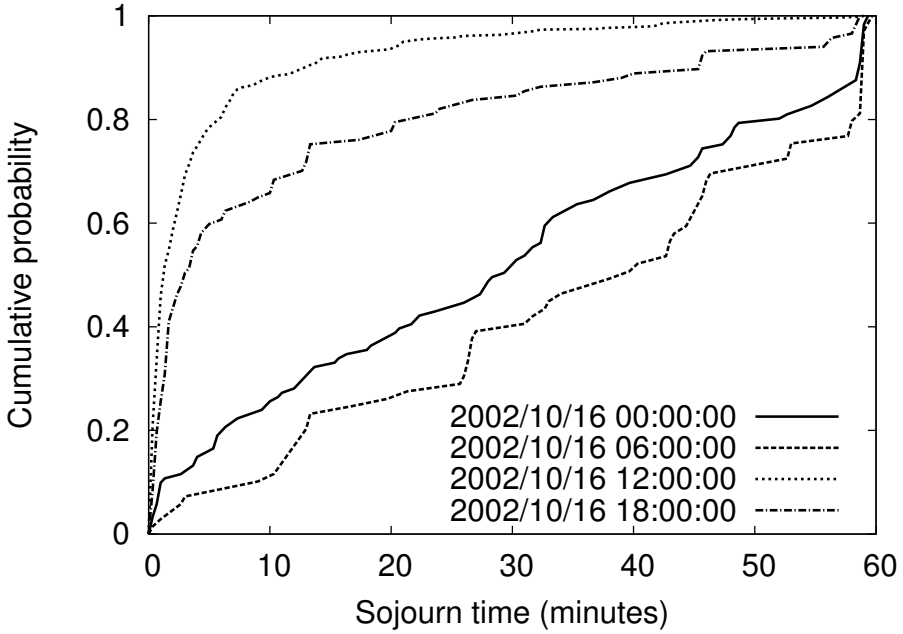


Figure 6.4: Cumulative distribution of the MUs' sojourn times

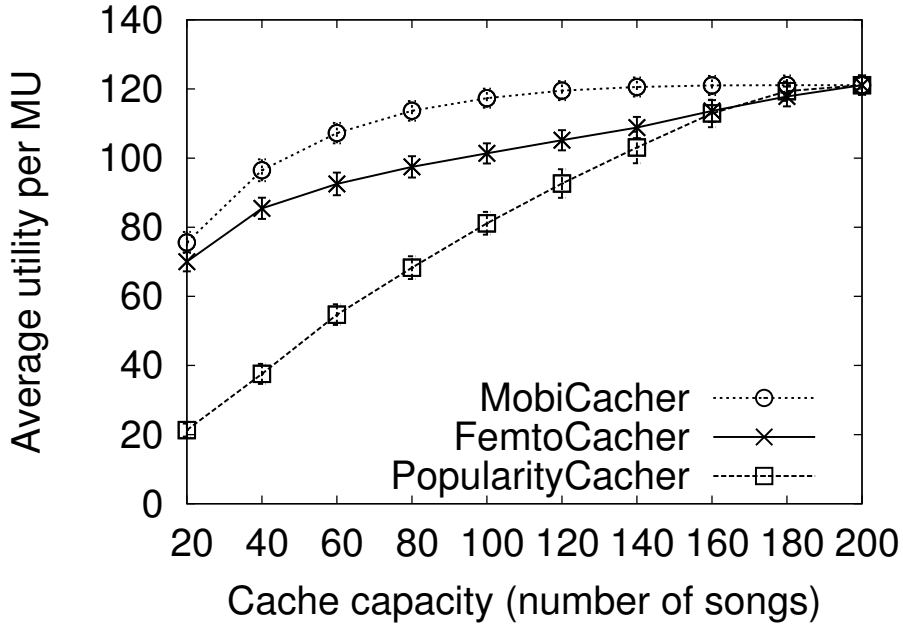


Figure 6.5: Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 00:00:00)

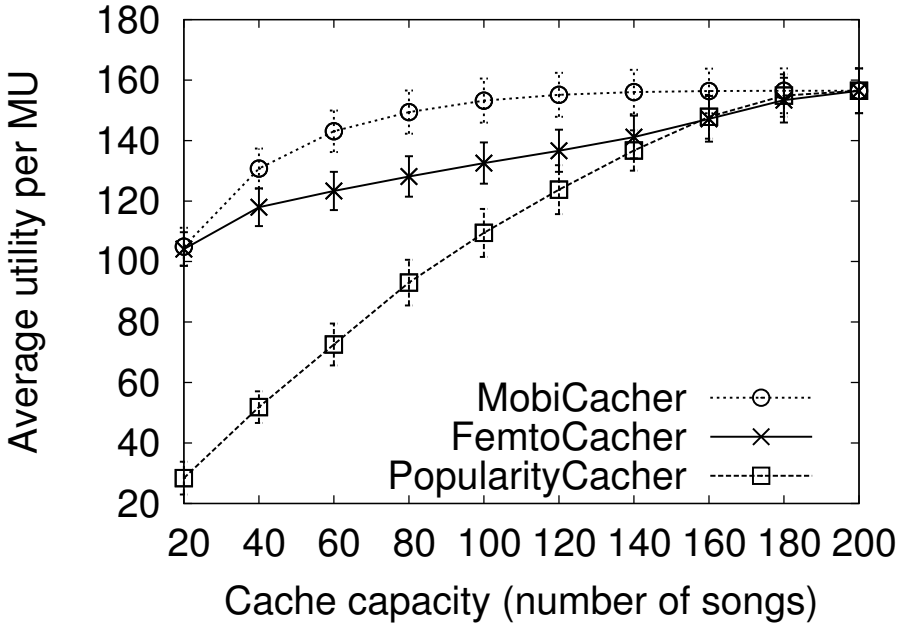


Figure 6.6: Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 06:00:00)

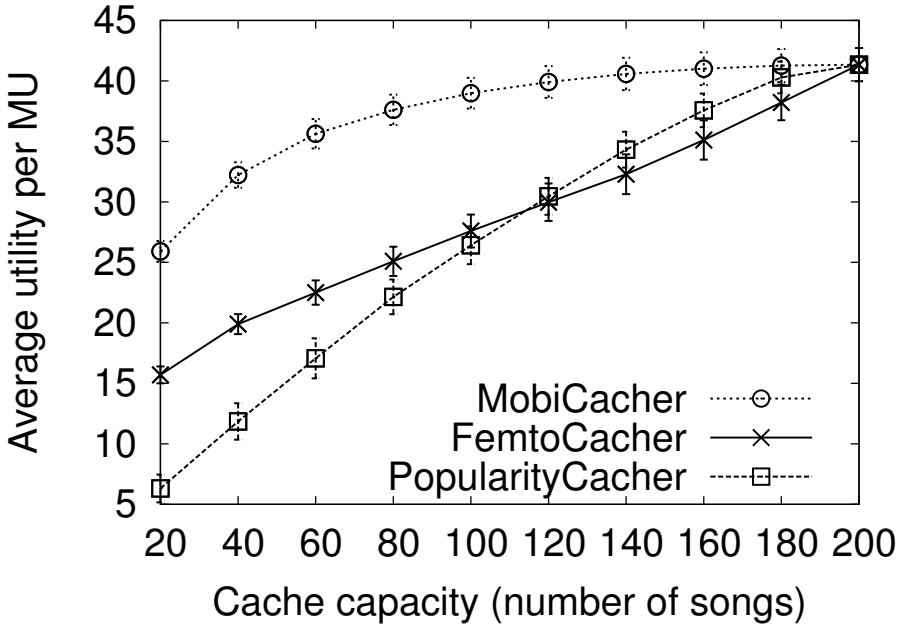


Figure 6.7: Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 12:00:00)

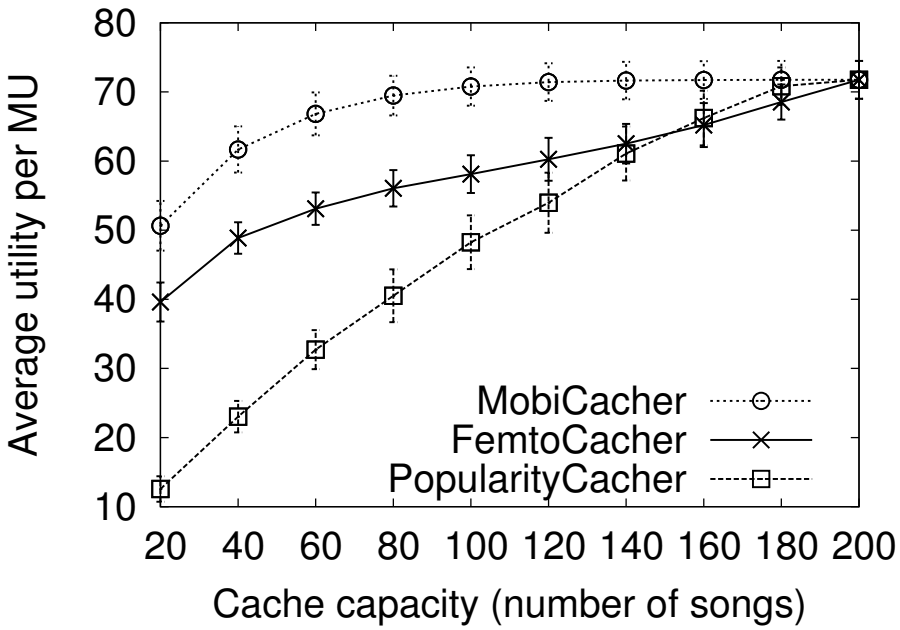


Figure 6.8: Impact of cache capacity at each small-BS (Simulation start time: 2002/10/16 18:00:00)

MUs' mobility is not considered and the caching decision is made based on the geographical distribution of MUs at the beginning of the simulations, and (3) *PopularityCacher* – each small-BS caches the most popular songs.

Figs. 6.5-6.8 plot the average caching utility gained by each MU against the cache capacity of each small-BS at different times during a day, respectively. The results clearly show that the caching utility produced by *MobiCacher* over the four one-hour simulation periods is higher than the utility generated by *FemtoCacher* and *PopularityCacher*. In particular, the performance improvement of *MobiCacher* is more significant when the cache capacity is small, *i.e.*, when each small-BS can cache less than 60 songs. This result is due to the fact that, when the storage capacity of each cache is small, *MobiCacher* can make more informed caching decisions by considering the mobility of MUs. On the contrary, when the cache capacity is large (*i.e.*, each small-BS can cache up to 160 songs), the three caching schemes perform similarly as almost the entire song library can be cached at each small-BS.

PopularityCacher produces the least utility compared with that of *MobiCacher* and *FemtoCacher*, as *PopularityCacher* only considers the popularity of the songs but not the

geographical distribution of MUs nor the mobilities of MUs. Also, a song that is (the most) popular globally may not be (the most) popular among the MUs associated with a particular small-BS. As all of the small-BSs store the most popular songs, the contents cached in the small-BSs at different locations are not diversified. In other words, the fact that an MU may access multiple small-BSs at a time is not taken advantage of, as the cached songs at each small-BS are the same.

Both *MobiCacher* and *FemtoCacher* consider the geographical distribution of MUs. *FemtoCacher* assumes that MUs are static and only looks at the initial MU distribution. In practice, an MU may move among the cells of small-BSs. As time progresses, the current geographical distribution of MUs can be quite different from their initial distribution. *FemtoCacher* fails to exploit this fact. The influence of MU mobility is shown in Figs. 6.9-6.12 where we plot the cumulative utility against the simulation time. We can see that *FemtoCacher* performs similarly or even slightly better than *MobiCacher* at the beginning of the simulation. But as time progresses and MUs start moving to different locations, the cumulative utility of *FemtoCacher* lags behind that of *MobiCacher*. The impact of MUs' mobility can also be observed from Figs. 6.5-6.8. When MUs are less mobile (*i.e.*, at night at 00:00:00 and in the early morning at 06:00:00), the gap between *MobiCacher* and *FemtoCacher* is relatively small. However, during the daytime when MUs are highly mobile, *MobiCacher* outperforms *FemtoCacher* by up to 27% as shown in Fig. 6.7 and Fig. 6.8.

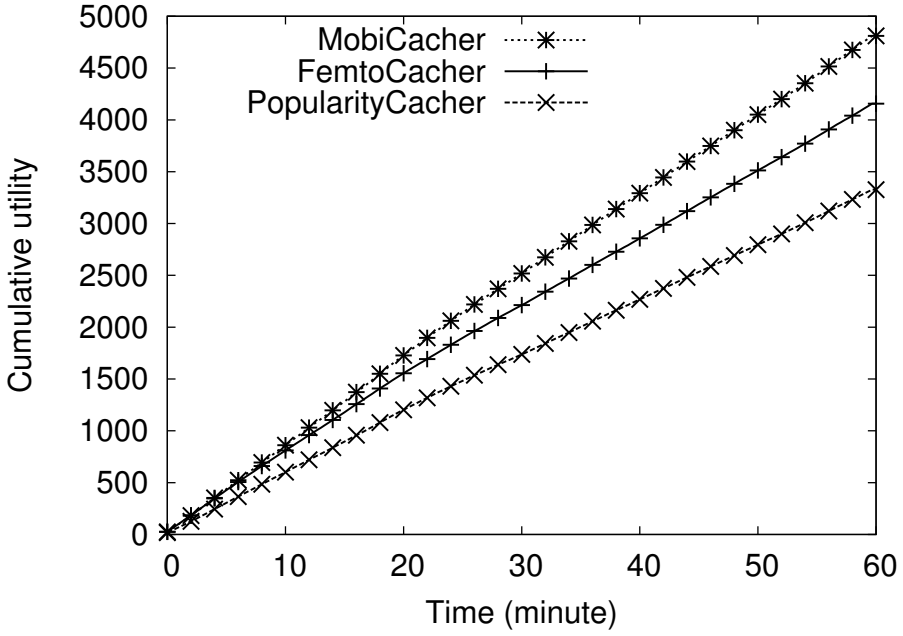


Figure 6.9: Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 00:00:00)

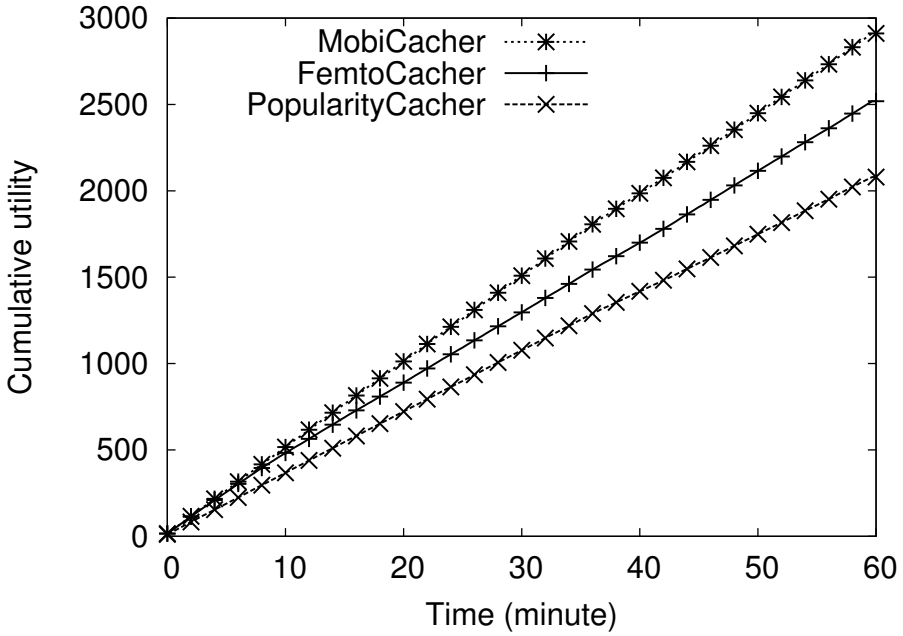


Figure 6.10: Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 06:00:00)

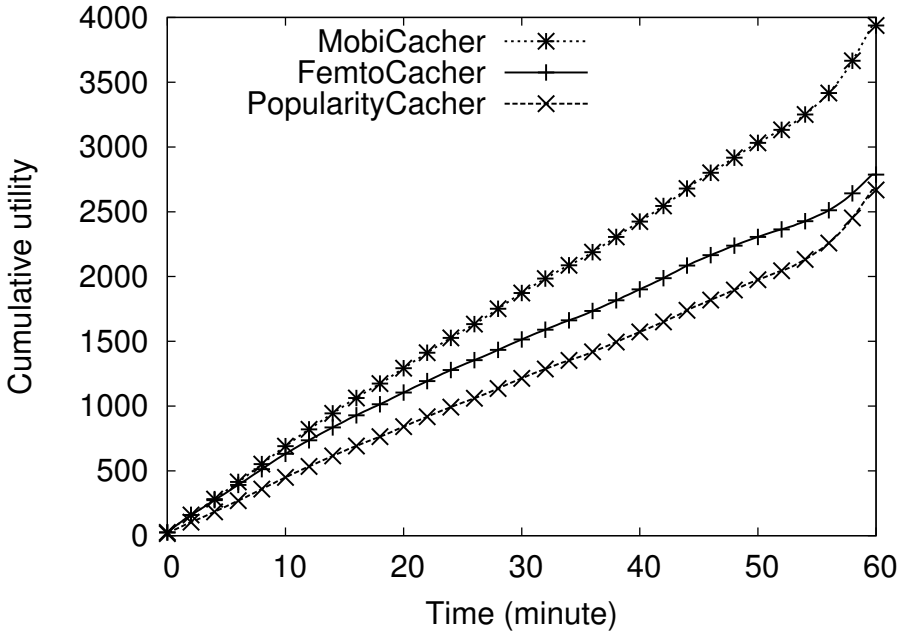


Figure 6.11: Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 12:00:00)

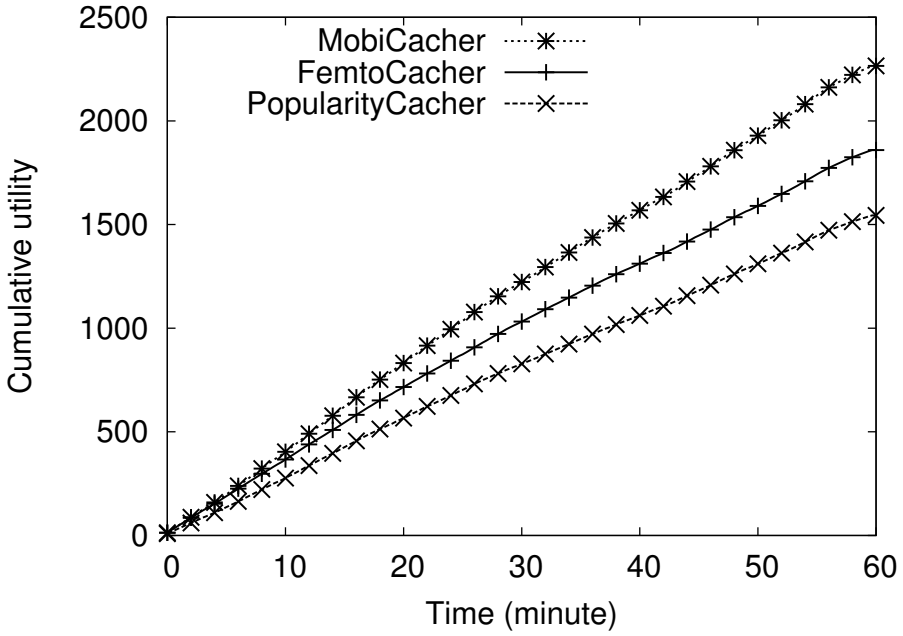


Figure 6.12: Cumulative utility against time (cache capacity: 100 songs, simulation start time: 2002/10/16 18:00:00)

6.7 Summary

The chapter defines a mobility-aware content caching problem, which caches popular contents at base stations of small-cell networks to ease the stress on backhaul networks. We formulate the problem into an optimization framework with the objective of maximizing the utility of caching. Given the NP-completeness of the problem, we developed the *MobiCacher* heuristic solution. We prove that the approximation ratio of *MobiCacher* is bounded and evaluate the performance via trace-based simulation. Results show that *MobiCacher* yields higher caching utility than other caching solutions, *i.e.*, by considering the mobility of MUs, *MobiCacher* can effectively improve caching efficiency and reduce the traffic/cost on backhaul networks.

Chapter 7

FUTURE WORK

7.1 Scheduling of QiF Charging Tours

The joint QiF charging tour planning and sensory data routing problem computes a set of QiF charging tours, along with their respective traveling distance and execution frequency. Given the execution frequency, each QiF charging tour needs to be executed *periodically* to charge the sensors covered by the tour. Each execution of a QiF charging tour is termed as an *execution instance* of the tour, involving the process that a QiF departs from the ES, travels to and charges the sensors along the tour, and returns to the ES.

7.1.1 Research Issues and Related Work

For each execution instance, the difference between the time that a QiF departs from the ES and the time that the QiF returns to the ES is termed as the *execution duration* of the execution instance, which equals the sum of the QiF's traveling time and the total charging time at the covered sensors. For a particular QiF charging tour, the traveling time remains the same for different execution instances, since the traveling distance is constant. However, the total charging time of one execution instance may be different from that of other instances. To further explain this fact, let us consider two consecutive execution instances of charging tour τ_1 in Fig. 7.1. Supposing the first execution instance of τ_1 ($T_{\tau_1}^j$) ends at time t, the second instance must be completed no later than $t + \frac{1}{F(\tau)}$, where $F(\tau_1)$ is the execution frequency of tour τ_1. In other words, the inverse of the execution frequency of a QiF charging tour specifies a *deadline* (from the moment when the 'current' execution instance ends) for the *latest* time when the 'next' execution instance of the charging tour must be completed. Given such a deadline, however, the execution duration of a charging tour's execution instance varies depending on when that execution instance starts before the deadline. When an execution

instance of a charging tour starts earlier, the covered sensors are left with more energy (at the time when a QiF arrives), which results in shorter charging time at the sensors and hence a shorter execution duration, even though the traveling time remains the same. An example is the $(j+2)$th instance of tour τ_1 ($T_{\tau_1}^{j+2}$) shown in Fig. 7.1, whose execution is started earlier than the deadline so that the execution duration becomes shorter (than $T_{\tau_1}^{\prime j+2}$). Notice that when one instance of a tour is completed earlier, the deadline for the tour's subsequent execution instances need to be moved forward in time, since the time interval between two consecutive execution instances cannot be longer than $\frac{1}{F(\tau)}$. For instance, the earlier start (or end) of instance $T_{\tau_1}^{j+2}$ *restarts* a 'new' periodic execution of tour τ_1 with the same period $\frac{1}{F(\tau_1)}$.

Given that multiple QiF charging tours are needed to cover all the sensors in the network, we need to schedule their (periodic) executions by QiFs. If we schedule every execution instance by strictly following the instance's deadline, it may be inevitable that, at certain times, two or more execution instances of different charging tours would overlap and we would need more than one QiF to execute these instances simultaneously. Nevertheless, if we start some execution instances earlier than their deadlines, the number of overlapped tours may be reduced to avoid deploying extra QiFs. For example, in Fig. 7.1, should the $(j+2)$th instance of tour τ_1 be executed just to meet the deadline (as represented by the empty dashed box) and overlap the $(k+3)$th instance of tour τ_2 in time, at least 2 QiFs will be needed to execute both charging tours at the same time. However, when we schedule τ_1's $(j+2)$th execution instance earlier (as represented by the shortened gray box) we will be able to avoid such conflicts and use only one QiF to execute both charging tours. Given a set of QiF charging tours computed by the joint tour planning and data routing, our objective is to schedule the executions of the computed QiF charging tours to minimize the number of QiF's needed.

Given the execution frequency of each QiF charging tour, the problem of scheduling the execution instances of QiF charging tours is closely related to the real-time scheduling of *periodic* tasks which has been extensively studied in solutions such as Rate Monotonic

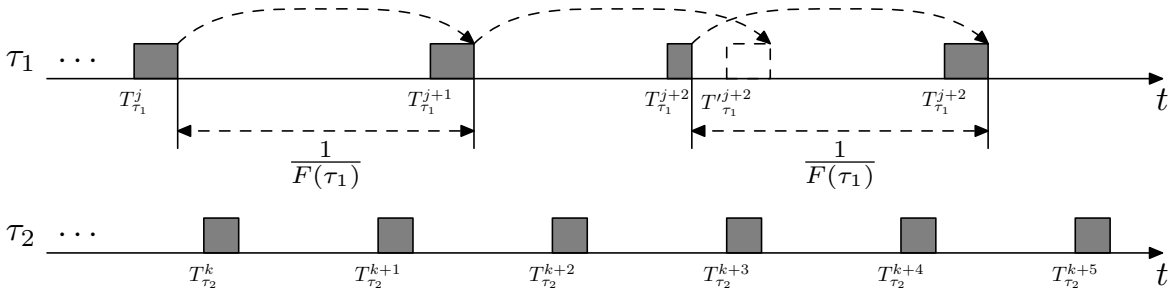


Figure 7.1: Executions of two QiF charging tours

(RM) scheduling [43], earliest-deadline-first scheduling, and many of their variants. In addition, [6] and [44] investigated the issue of RM scheduling on *multiprocessors* and derived schedulability tests based on the utilization bound for RM scheduling on multiprocessors with partitioning-based scheduling. As described above, the problem of scheduling the execution instances of QiF charging tours is fundamentally different from the periodic tasks assumed in RM scheduling. Specifically, each execution instance is not preemptible; the execution duration of an execution instance is shortened when the instance starts earlier; and the earlier start of an execution instance *restarts* a 'new' periodic task (with the same frequency/period).

The design of a specific tour scheduling algorithm needs to determine the following aspects for each execution instance: (1) when will an execution instance of a QiF charging tour start (*i.e.*, when will a QiF depart from the ES); (2) when will a QiF arrive at a particular sensor; (3) when will a QiF depart from a particular sensor; (4) when will an execution instance end (*i.e.*, when does a QiF return to the ES). To fully investigate these factors, both the QiF's traveling time (assumed to be zero in [42]) between sensors and the charging time at each sensor (assumed to be zero in [91]) must be considered. Moreover, when a QiF returns to the ES after completing an execution instance, the battery of the QiF may be low or even empty. Thus, the QiF has to stay at the ES to replenish its battery, and cannot execute another charging tour immediately. Therefore, the time needed to charge QiF's battery must also be considered. Also, the speed of QiFs has a significant impact on the execution duration of an execution instance of a charging tour. Depending on the energy model of QiF's movement, there is a clear tradeoff between execution duration and energy consumed during travel.

7.1.2 Proposed Work

We plan to study two types of algorithms for the QiF charging tour scheduling problem: *partitioned* scheduling algorithm and *global* scheduling algorithm [13]. In the partitioned algorithm, the given set of QiF charging tours are partitioned into several disjoint subsets such that all the execution instances of the tours in the same subset are to be executed by the same dedicated QiF. Two issues arise. The first issue is the *schedulability* with a single QiF, which decides whether a single QiF could execute a given subset of tours. With such a schedulability test, the second issue is how a given set of QiF charging tours should be partitioned so that tours in each subset can be executed by a single QiF and the number of subsets (and hence QiFs) is kept minimal. In contrast, in the global algorithm, different execution instances of the same charging tour may be executed by different QiFs. Now, the question becomes the schedulability with K QiFs, given a set of charging tours. Clearly, the global algorithm is more challenging than the partitioned algorithm, but investigating the schedulability with a single QiF will provide insight to the study of schedulability with K QiFs. The schedulability problem is further complicated by the stochastic nature of charging tours when energy consumption rates become probabilistic. We plan to systematically investigate this issue and develop sound solutions.

7.2 Dynamic Content Caching with Mobility Prediction

In Chapter 6, we have shown that *MobiCacher* improves caching utility in small-cell networks by considering the mobility patterns of MUs. In the discussion, we assumed that the trajectories of MUs are either known *a priori* or can be accurately predicted. Also, we assumed that the MUs' preferences for contents are constant and a cached content will never be replaced by another content, *i.e.*, no cache replacement. We plan to lift these assumptions and develop a caching solution for more versatile scenarios. We discuss relevant issues in Sections 7.2.1-7.2.3.

7.2.1 Predicting MU Mobility

In practice, although the mobility patterns of MUs may not be perfectly known, they can be predicted by mobility prediction algorithms. However, prediction error is inherent in any mobility prediction algorithm. We plan to investigate the content caching problem together with a mobility prediction algorithm and evaluate the impact of prediction error.

In particular, we plan to use the second-order Markov model [52, 72] as the mobility predictor. The second-order Markov model is built from the mobility history h of an MU, which consists of a sequence $X_1X_2\cdots X_n$ of previous states. Each X_i in h records the MU's state in time slot i, and can be specified either by the MU's location or by the set of small-BSs that the MU can sense in time slot i. The second-order Markov model predicts an MU's state X_{n+1} in time slot $n+1$ based on the nearest two previous states of the MU, *i.e.*, X_{n-1} and X_n. Specifically, given $X_{n-1}=s_i$ and $X_n=s_j$, the probability of $X_{n+1}=s_k$ is predicted as

$$\mathrm{P}(X_{n+1}=s_k|X_{n-1}=s_i,X_n=s_j)=\frac{N(s_is_js_k,h)}{N(s_is_j,h)}, \tag{7.1}$$

where $N(s_is_js_k,h)$ and $N(s_is_j,h)$ are the number of times that the sub-sequences $s_is_js_k$ and s_is_j occur in the mobility history h, respectively. Fig. 7.2 shows an example of a second-order Markov model. Using (7.1), the Markov model can either output a state with the highest probability, *i.e.*,

$$\operatorname*{argmax}_{s_k}\mathrm{P}(X_{n+1}=s_k|X_{n-1}=s_i,X_n=s_j),$$

as the predicted state of the MU in time slot $n+1$, or generate a probability distribution of all possible states in time slot $n+1$.

7.2.2 Cache Replacement Policy

We also plan to develop a cache replacement policy based on the mobility predictor. Specifically, when receiving a request for an uncached content, a small-BS retrieves the content over the backhaul from a remote server. Upon the arrival of the content, the small-BS

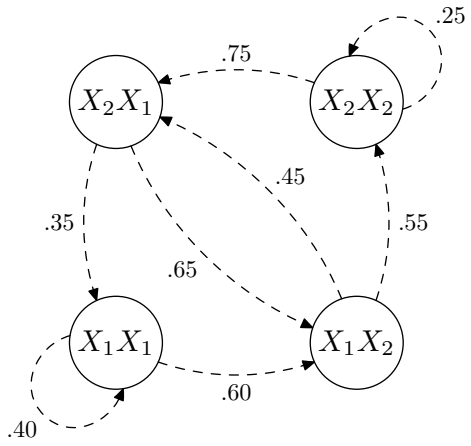


Figure 7.2: An example of a second-order Markov mobility predictor

calculates a likelihood that the content will be requested again in the future, considering the results calculated by the mobility predictor as well as the MUs' preferences to the content. Based on the calculated likelihood, the small-BS can make a decision on whether the new content should be discarded or be used to replace an existing content in the cache.

7.2.3 Time-Variant User Preferences

It is also assumed in Chapter 6 that MUs' preferences for contents are known *a priori* and remain constant. Following the static preference distribution, an MU makes requests to the contents at different times and may request the same content repeatedly. We plan to incorporate a more realistic model for user preferences. First, the MUs' preferences for contents should be time-variant. For example, an MU's preference for a particular content may become the highest in a short time right after the creation of the content, and then the preference fade away gradually over time. Also, an MU's preference for a newly generated content may be predicted based on the MU's preferences to similar contents by analyzing the MU's play history. Last, an MU's preference for certain types of contents may diminish drastically after downloading the content for the first time, and the MU may no longer request the content in the future. For example, an MU might only download and watch a news video once.

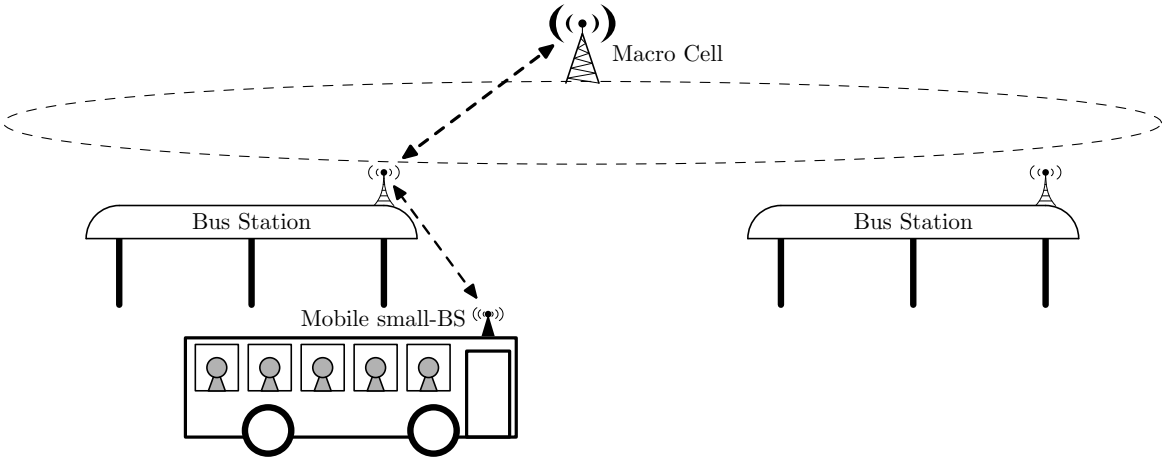


Figure 7.3: Network architecture for mobile small-BSs

7.3 Content Caching at Mobile Small-Cell Base Station with Group Mobility

The growing popularity of Internet-capable mobile devices also stipulates the demand for Internet access on high-speed vehicles such as buses, subways and trains. To meet such demand, the idea of mobile small-cell base stations (mobile small-BSs[1]) has been proposed to extend ubiquitous and affordable Internet service into moving vehicles [68, 55]. Fig. 7.3 depicts a typical network architecture for mobile small-BSs where a group of in-vehicle users are connected to a mobile small-BS mounted on the vehicle. The mobile small-BS relays the traffic from/to the users either through a road-side base station (*e.g.*, mounted at a bus station) or through a macro-cell base station.

Compared to direct device-to-macro-cell communications, mobile small-BSs provide several benefits [54]. For example, mobile small-BSs provide higher data rates because the mobile small-BSs can be equipped with sizable antennas, yielding better communication channels between mobile small-BSs and macro-cell base stations. Also, a mobile small-BS is less energy constrained than the users' mobile devices. Moreover, the trajectory and schedule of buses and trains are highly predictable, enabling simple and efficient handoff management.

Given the benefits of mobile small-BSs, we propose to study the problem of content

[1] Also known as *mobile hotspots*.

caching at mobile small-BSs. The problem is further motivated by the fact that the preferences of users moving together may be highly correlated. For example, most morning bus riders may be highly interested in the news of the day. Thus the locality of the requested contents can be better explored under such *group mobility*. Also, the knowledge of the mobility trajectory of a mobile small-BS can facilitate intelligent caching decision. For instance, given the length of contact time between a mobile small-BS and a road-side base station, as well as the link capacity, a caching algorithm can plan ahead what contents and how many contents the mobile small-BS should download from the road-side base station. Moreover, the road-side base station can potentially pre-fetch and buffer the contents to be requested, further reducing the content retrieval delay.

7.3.1 Preliminary Study

We present a preliminary formulation for the problem of content caching at mobile small-BSs. As a first step, we assume a mobile small-BS can only communicate with road-side base stations installed at bus stops. We use an ordered set $\mathcal{S} = \{1, \cdots, S\}$ to denote the sequence of stops at which the bus stops along the its route. At each stop, the mobile small-BS can download new contents to replace the existing ones in the cache. We use C_i to denote the set of contents cached by the mobile small-BS upon the bus leaving stop i. C_i is assumed to remain constant when the bus is in route from stop i to stop $i+1$.

A set of riders are denoted by $\mathcal{R}$. Each rider $r \in \mathcal{R}$ is specified in terms of $b_r \in \mathcal{S}$, the stop at which the rider gets on the bus, and $o_r \in \mathcal{S}$, the stop at which the rider gets off the bus. Each rider may have different preferences for contents, and we use $c_{r,l}$ to denote rider r's preference for content l. Our goal is to maximally satisfy the riders' interests by properly populating the cache of the mobile small-BS at different times. The caching utility is defined directly based on the riders' preferences. Specifically, if content l is cached during the interval when rider r is on the bus, the system gains an utility of $c_{r,l}$. Given b_r and o_r of rider r, the set of cached contents that can be accessed by rider r can be expressed as $\cup_{b_r \le i < o_r} C_i$, and the total caching utility gained from rider r is $\sum_{l \in \cup_{b_r \le i < o_r} C_i} c_{r,l}$. The total

caching utility $\mathcal{U}$ is defined as the summation of utility gained from every rider as follows.

$$\mathcal{U} = \sum_{r \in \mathcal{R}} \sum_{l \in \cup_{b_r \leq i < o_r} C_i} c_{r,l} \tag{7.2}$$

The problem of content caching at mobile small-BSs is now formulated as follows.

$$\max \quad \mathcal{U} = \sum_{r \in \mathcal{R}} \sum_{l \in \cup_{b_r \leq i < o_r} C_i} c_{r,l} \tag{7.3}$$

$$\text{s.t.} \quad C_i \in \mathcal{L} \qquad \forall i \in \mathcal{S} \tag{7.4}$$

$$|C_i| \leq M \qquad \forall i \in \mathcal{S} \tag{7.5}$$

The above formulation maximizes the caching utility (7.3). Eq. (7.4) ensues all cached contents are from a given library $\mathcal{L}$ of contents and (7.5) guarantees that at any time, the number of cached contents does not exceed the caching capacity M.

BIBLIOGRAPHY

[1] G. Açar and A. E. Adams. ACMENet: an underwater acoustic sensor network protocol for real-time environmental monitoring in coastal areas. *IET Proc. Radar, Sonar and Navigation*, 153(4):365–380, Aug. 2006.

[2] I. F. Akyildiz, D. Pompili, and T. Melodia. Underwater acoustic sensor networks: research challenges. *Elsevier Ad Hoc Networks*, 3(3):257–279, Mar. 2005.

[3] I. Ashraf, F. Boccardi, and L. Ho. SLEEP mode techniques for small cell deployments. *IEEE Commun. Mag.*, 49(8):72–79, Aug. 2011.

[4] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proc. ACM Internet Measurement Conf. (IMC)*, Nov. 2009.

[5] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine. Relays, base stations, and meshes: Enhancing mobile networks with infrastructure. In *Proc. ACM Int'l Conf. on Mobile Computing and Networking (MobiCom)*, Sept. 2008.

[6] S.K. Baruah and J. Goossens. Rate-monotonic scheduling on uniform multiprocessors. *IEEE Trans. Computers*, 52(7):966–970, May 2003.

[7] T. Bertin-Mahieux, D. P.W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proc. Int'l Society for Music Information Retrieval Conf. (ISMIR)*, Oct. 2011.

[8] C. S. Bontu and E. Illidge. DRX mechanism for power saving in LTE. *IEEE Commun. Mag.*, 47(6):48–55, Jun. 2009.

[9] M. Cardei and D.Z. Du. Improving wireless sensor network lifetime through power aware organization. *ACM Wireless Networks*, 11(3):333–340, May 2005.

[10] Concorde TSP Solver. `http://www.tsp.gatech.edu/concorde.html`.

[11] A. Damnjanovic, J. Montojo, Y. Wei, T. Ji, T. Luo, M. Vajapeyam, T. Yoo, O. Song, and D. Malladi. A survey on 3GPP heterogeneous networks. *IEEE Wireless Commun.*, 18(3):10–21, Jun. 2011.

[12] S. K. Dandapat, S. Pradhan, N. Ganguly, and R. R. Choudhury. Sprinkler: Distributed content storage for just-in-time streaming. In *Proc. ACM Workshop on Cellular Networks: Operations, Challenges, and Future Design (CellNet)*, Jun. 2013.

[13] R. I. Davis and A. Burns. A survey of hard real-time scheduling for multiprocessor systems. *ACM Computing Surveys*, 43(4), Oct. 2011.

[14] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire. Femtocaching: Wireless video content delivery through distributed caching helpers. In *Proc. IEEE Int'l Conf. Computer Commun. (INFOCOM)*, Mar. 2012.

[15] Y. Guan, C.C. Shen, and J. Yackoski. MAC scheduling for high throughput underwater acoustic networks. In *Proc. IEEE Wireless Commun. and Networking Conf. (WCNC)*, Mar. 2011.

[16] Y. Guan, Y. Xiao, L. J. Cimini, and C.C. Shen. Power efficient peer-to-peer streaming to co-located mobile users. In *Proc. IEEE Consumer Commun. and Networking Conf. (CCNC)*, Jan. 2014.

[17] Y. Guan, Y. Xiao, H. Feng, C.C. Shen, and L. J. Cimini. MobiCacher: mobility-aware content caching in small-cell networks. In *Proc. Global Commun. Conf. (GLOBECOM)*, Dec. 2014. submitted for publication.

[18] Y. Guan, Y. Xiao, C.C. Shen, and L. J. Cimini. CSR: Cooperative source routing using virtual MISO in wireless ad hoc networks. In *Proc. IEEE Wireless Commun. and Networking Conf. (WCNC)*, Mar. 2011.

[19] M. Haenggi and D. Puccinelli. Routing in ad hoc networks: a case for long hops. *IEEE Commun. Mag.*, 43(10):93–101, Oct. 2005.

[20] B. Han, P. Hui, and A. Srinivasan. Mobile data offloading in metropolitan area networks. *ACM Mobile Comput. Commun. Rev.*, 14(4):28–30, Nov. 2010.

[21] B. He, B. Xie, and D. P. Agrawal. Optimizing deployment of internet gateway in wireless mesh networks. *Elsevier Computer Commun.*, 31(7):1259–1275, May 2008.

[22] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li. Research challenges and applications for underwater sensor networking. In *Proc. IEEE Wireless Commun. and Networking Conf. (WCNC)*, Apr. 2006.

[23] W. B. Heinzelman. *Application-specific protocol architectures for wireless networks*. PhD thesis, Massachusetts Institute of Technology, 2000.

[24] J. Hoydis, M. Kobayashi, and M. Debbah. Green small-cell networks. *IEEE Vehicular Technology Mag.*, 6(1):37–43, Mar. 2011.

[25] C.C. Hsu, K.F. Lai, C.F. Chou, and K.C.J. Lin. ST-MAC: Spatial-Temporal MAC Scheduling for Underwater Sensor Networks. In *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2009.

[26] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4G LTE networks. In *Proc. ACM Int'l Conf. Mobile Systems, Applicat., and Services (MobiSys)*, Jun. 2012.

[27] Cisco Visual Networking Index. Global mobile data traffic forecast update, 2013 - 2018. *White Paper*, Feb. 2013.

[28] G. Jakllari, S.V. Krishnamurthy, M. Faloutsos, P.V. Krishnamurthy, and O. Ercetin. A cross-layer framework for exploiting virtual MISO links in mobile ad hoc networks. *IEEE Tran. Mobile Computing*, 6(6):579–594, Jun. 2007.

[29] X. Jin and Y.K. Kwok. Cloud assisted P2P media streaming for bandwidth constrained mobile subscribers. In *Proc. IEEE Int'l Conf. on Parallel and Distributed Systems (ICPADS)*, Dec. 2010.

[30] R.K. Kalle, A.K. Nandan, and D. Das. La VoLTE: novel cross layer optimized mechanism of video transmission over LTE for DRX. In *Proc. IEEE Vehicular Technology Conf. (VTC)*, May 2012.

[31] A. Kansal, J. Hsu, S. Zahedi, and M.B. Srivastava. Power management in energy harvesting sensor networks. *ACM Trans. Embedded Computing Systems*, 6(4):32–69, Sept. 2007.

[32] A. Khandani, E. Modiano, J. Abounadi, and L. Zheng. Cooperative routing in wireless networks. In *Advances in Pervasive Computing and Networking*, pages 97–117. Springer, 2005.

[33] R. Kumar, Y. Liu, and K. Ross. Stochastic fluid theory for P2P streaming systems. In *Proc. IEEE Int'l Conf. Comput. Commun. (INFOCOM)*, May 2007.

[34] J. F. Kurose. *Computer Networking: A Top-Down Approach Featuring the Internet, 5/E*. Pearson Education, 2009.

[35] A. Kurs, A. Karalis, R. Moffatt, J.D. Joannopoulos, P. Fisher, and M. Soljačić. Wireless power transfer via strongly coupled magnetic resonances. *Science*, 317(5834):83–86, Jul. 2007.

[36] S. Lakshmanan and R. Sivakumar. Diversity routing for multi-hop wireless networks with cooperative transmissions. In *Proc. IEEE Conf. Sensor, Mesh and Ad Hoc Commun. and Networks (SECON)*, Jun. 2009.

[37] J.N. Laneman and G.W. Wornell. Distributed space-time-coded protocols for exploiting cooperative diversity in wireless networks. *IEEE Trans. Information Theory*, 49(10):2415–2425, Oct. 2003.

[38] G. Laporte, M. Desrochers, and Y. Nobert. Two exact algorithms for the distance-constrained vehicle routing problem. *Networks*, 14(1):161–172, 1984.

[39] X. Li, M. Dong, Z. Ma, and F. C.A. Fernandes. GreenTube: power optimization for mobile video streaming via dynamic cache management. In *Proc. ACM Int'l Conf. Multimedia (MM)*, Oct. 2012.

[40] Y. Li, G. Su, P. Hui, D. Jin, L. Su, and L. Zeng. Multiple mobile data offloading through delay tolerant networks. In *Proc. ACM Workshop on Challenged Networks (CHANTS)*, Sept. 2011.

[41] Z. Li, B. Li, D. Jiang, and L. C. Lau. On achieving optimal throughput with network coding. In *Proc. IEEE Int'l Conf. Comput. Commun. (INFOCOM)*, Mar. 2005.

[42] Z. Li, Y. Peng, W. Zhang, and D. Qiao. Study of joint routing and wireless charging strategies in sensor networks. In *Proc. Springer Int'l Conf. Wireless Algorithms, Systems, and Applicat. (WASA)*, Aug. 2010.

[43] C. L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1):46–61, Jan. 1973.

[44] J. M. Lopez, J. L. Diaz, and D. F. Garcia. Minimum and maximum utilization bounds for multiprocessor rate monotonic scheduling. *IEEE Trans. Parallel and Distributed Systems*, 15(7):642–653, Jul. 2004.

[45] D. Lopez-Perez, I. Guvenc, G. De La Roche, M. Kountouris, T. Q. S. Quek, and J. Zhang. Enhanced intercell interference coordination challenges in heterogeneous networks. *IEEE Wireless Commun.*, 18(3):22–30, Jun. 2011.

[46] H. Luo, X. Meng, R. Ramjee, P. Sinha, and L. Li. The design and evaluation of unified cellular and ad hoc networks. *IEEE Trans. Mobile Computing*, 6(9):1060–1074, Sept. 2007.

[47] K.J. Ma, R. Bartos, S. Bhatia, and R. Nair. Mobile video delivery with HTTP. *IEEE Commun. Mag.*, 49(4):166–175, Apr. 2011.

[48] A. Makhorin. GLPK (GNU linear programming kit), 2008.

[49] M. McNett and G. M. Voelker. Access and mobility of wireless PDA users. *ACM SIGMOBILE Mobile Computing and Commun. Review*, 9(2):40–55, Apr. 2005.

[50] Y. Mei, Y. Hua, A. Swami, and B. Daneshrad. Combating synchronization errors in cooperative relays. In *Proc. IEEE Int'l Conf. on Acoust., Speech, and Signal Process. (ICASSP)*, Mar. 2005.

[51] Y. Mei, Y.H. Lu, Y.C. Hu, and C.S.G. Lee. Deployment of mobile robots with energy and timing constraints. *IEEE Trans. Robotics*, 22(3):507–522, 2006.

[52] A. J. Nicholson and B. D. Noble. Breadcrumbs: Forecasting mobile connectivity. In *Proc. ACM Int'l Conf. Mobile Computing and Networking (MobiCom)*, Sept. 2008.

[53] N. B. Niraula, K. Kanchanasut, and A. Laouiti. Peer-to-peer live video streaming over mobile ad hoc network. In *Proc. Int'l Conf. Wireless Commun. and Mobile Computing (IWCMC)*, Jun. 2009.

[54] S. Pack, H. Rutagemwa, X. Shen, J. W. Mark, and L. Cai. Performance analysis of mobile hotspots with heterogeneous wireless links. *IEEE Tran. Wireless Commun.*, 6(10), Oct. 2007.

[55] S. Pack, X. Shen, J. W. Mark, and J. Pan. Mobility management in mobile hotspots with heterogeneous multihop wireless links. *IEEE Commun. Mag.*, 45(9):106 – 112, Sept. 2007.

[56] J. Pang, B. Greenstein, M. Kaminsky, D. McCoy, and S. Seshan. WiFi-reports: Improving wireless network selection with collaboration. *IEEE Trans. Mobile Computing*, 9(12):1713–1731, Dec. 2010.

[57] A. Paulraj, R. Nabar, and D. Gore. *Introduction to space-time wireless communications*. Cambridge University Prress, 2003.

[58] Y. Peng, Z. Li, W. Zhang, and D. Qiao. Prolonging sensor network lifetime through wireless charging. In *Proc. IEEE Real-Time Systems Symposium (RTSS)*, Nov. 2010.

[59] PPTV. http://www.pptv.com.

[60] R. Prasad and H. Wu. Gateway deployment optimization in cellular wi-fi mesh networks. *Journal of Networks*, 1(3):31–39, 2006.

[61] F. Qiu, J. Bai, Y. Cui, and Y. Xue. Optimal rate allocation in peer-to-peer streaming over wireless networks. In *Proc. Int'l Conf. Collaboration Technologies and Systems (CTS)*, May 2011.

[62] QualNet Simulator. http://www.scalable-networks.com.

[63] A. Rahmati and L. Zhong. Context-for-wireless: context-sensitive energy-efficient wireless data transfer. In *Proc. ACM Int'l Conf. Mobile Systems, Applications and Services (MobiSys)*, Jun. 2007.

[64] R. Roverso, S. El-Ansary, and S. Haridi. Peer2View: a peer-to-peer HTTP-live streaming platform. In *Proc. IEEE Int'l Conf. on Peer-to-Peer Computing (P2P)*, Sept. 2012.

[65] Samsung. Converging telecom & it in the lte ran. *White paper*, 2013.

[66] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity. Part I. System description. *IEEE Trans. Commun.*, 51(11):1927–1938, Nov. 2003.

[67] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity. Part II. Implementation Aspects and Performance Analysis. *IEEE Trans. Commun.*, 51(11):1939–1948, Nov. 2003.

[68] A. Sharma, V. Navda, R. Ramjee, V. N. Padmanabhan, and E. M. Belding. Cool-Tether: energy efficient on-the-fly wifi hot-spots using mobile phones. In *Proc. ACM Int'l Conf. Emerging Networking Experiments and Technologies (CoNEXT)*, Dec. 2009.

[69] Y. Shi, L. Xie, Y. T. Hou, and H. D. Sherali. On renewable sensor networks with wireless energy transfer. In *Proc. IEEE Int'l Conf. Comput. Commun. (INFOCOM)*, Apr. 2011.

[70] S. Singh, O. Oyman, A. Papathanassiou, D. Chatterjee, and J. G. Andrews. Video capacity and QoE enhancements over LTE. In *Proc. IEEE Int'l Conf. Commun. (ICC)*, Jun. 2012.

[71] S. Slijepcevic and M. Potkonjak. Power efficient organization of wireless sensor networks. In *Proc. IEEE Int'l Conf. Commun. (ICC)*, Jun. 2001.

[72] L. Song, D. Kotz, R. Jain, and X. He. Evaluating location predictors with extensive Wi-Fi mobility data. In *Proc. IEEE Int'l Conf. Comput. Commun. (INFOCOM)*, Mar. 2004.

[73] H. Soroush, K. Sung, E. Learned-Miller, B. N. Levine, and M. Liberatore. Disabling GPS is Not Enough: Cellular location leaks over the Internet. In *Proc. Privacy Enhancing Technologies Symposium (PETS)*, Jul. 2013.

[74] Spotify. https://www.spotify.com.

[75] M. Stiemerling and S. Kiesel. A system for peer-to-peer video streaming in resource constrained mobile environments. In *Proc. ACM Workshop on User-provided Networking: Challenges and Opportunities (U-NET)*, Dec. 2009.

[76] A. A. Syed, W. Ye, J. Heidemann, and B. Krishnamachari. Understanding spatio-temporal uncertainty in medium access with aloha protocols. In *Proc. ACM Workshop on Underwater Networks (WuWNet)*, Sept. 2007.

[77] C.K. Toh. *Ad hoc mobile wireless networks: protocols and systems*. Pearson Education, 2001.

[78] B. Tong, Z. Li, G. Wang, and W. Zhang. How wireless power charging technology affects sensor network deployment and routing. In *Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS)*, Jun. 2010.

[79] P. Toth and D. Vigo. *The vehicle routing problem*, volume 9. Society for Industrial Mathematics, 2002.

[80] WatchESPN. http://espn.go.com/watchespn/about.

[81] W. Wayne. *Introduction to Mathematical Programming: Applications and Algorithms*. Duxbury Resource Center, 2003.

[82] Wireless Power Consortium. http://www.wirelesspowerconsortium.com.

[83] C. Wu and B. Li. Optimal peer selection for minimum-delay peer-to-peer streaming with rateless codes. In *Proc. ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming (P2PMMS)*, Nov. 2005.

[84] Y. Xiao, Y. Guan, W. Chen, C.C. Shen, and L. J. Cimini. Location-aware cooperative routing in multihop wireless networks. In *Proc. IEEE Wireless Commun. and Networking Conf. (WCNC)*, Mar. 2011.

[85] Y. Xiao, Y. Guan, L. J. Cimini, and C.C. Shen. Power efficient user pairing for multicasting in heterogeneous wireless networks. In *Proc. IEEE Military Commun. Conf. (MILCOM)*, Nov. 2013.

[86] G. Xie and J. Gibson. A networking protocol for underwater acoustic networks. Technical report, DTIC Document, 2000.

[87] J. Yackoski and C.C. Shen. UW-FLASHR: Achieving high channel utilization in a time-based acoustic MAC protocol. In *Proc. ACM Int'l Workshop on Underwater Networks (WuWNet)*, Sept. 2008.

[88] J. Yackoski, L. Zhang, B. Gui, C.C. Shen, and L. J. Cimini. Networking with cooperative communications: holistic design and realistic evaluation. *IEEE Commun. Mag.*, 47(8), Aug. 2009.

[89] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proc. IEEE Int'l Conf. Computer Commun. (INFOCOM)*, Jun. 2002.

[90] B. Zhang, S.H. G. Chan, G. Cheung, and E. Y. Chang. LocalTree: an efficient algorithm for mobile peer-to-peer live streaming. In *Proc. IEEE Int'l Conf. Commun. (ICC)*, Jun. 2011.

[91] M. Zhao, J. Li, and Y. Yang. Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks. In *Proc. IEEE Int'l Teletraffic Congress (ITC)*, Sept. 2011.