A COMMUNICATION FRAMEWORK FOR OPPORTUNISTIC MOBILE NETWORKS WITH DIVERSE CONNECTIVITY

A Dissertation

by

CHEN YANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Radu Stoleru
Committee Members,	Riccardo Bettati
	Anxiao Jiang
	Alexander Sprintson
Head of Department,	Dilma Da Silva

May 2018

Major Subject: Computer Engineering

ABSTRACT

An Opportunistic Mobile Network (OMN) refers to the network paradigm where wireless devices communicate with each other through the opportunistically formed wireless links. Routing in OMN relies on node mobility and the store-and-forward mechanism. It is paramount to have *energy efficient*, *robust* and *cost effective* routing protocols in such environments. Previous research usually assumes that the connectivity in such networks is *extremely sparse* and that the network is purely *infrastructure-less*. However, real world deployments of OMNs actually exhibit *diverse* connectivity, i.e., connectivity may range from sparsely connected to well connected or the network may coexist with infrastructure. Consequently, the simplified assumptions of previous solutions lead to suboptimal behaviors of routing protocols, which includes *redundant transmissions*, *too much or insufficient data replications*, *poor forwarding decisions*, etc.

In this dissertation, in order to address the aforementioned problems, we propose a communication framework for OMNs with diverse connectivity, which consists of a series of algorithms and protocols that aim to provide *energy efficient*, *robust* and *cost-aware* communication services to applications. In this framework, we propose: a) algorithms that carefully schedule transmissions in an opportunistic contact involving multiple nodes; b) routing protocols that consider simultaneously mobile nodes' delivery capability and traffic load; c) mathematical tools that characterize not only Inter-Contact Times but also their correlations; d) adaptive mechanisms to realize dynamic data replication; and e) forwarding strategies that optimally trade-off energy consumption and delay in a cost-aware fashion when utilizing infrastructure. We evaluate the proposed routing protocols and algorithms through extensive simu-

lations using both synthetic network models and real world mobility traces. We also conduct real world experiments on a wireless testbed to demonstrate their practicability. The evaluation results show that, with the assumption of diverse connectivity in mind, the proposed algorithms and protocols greatly improve the networking performance and efficiency. The consideration of delay correlations and a mechanism for dynamic replication are critical for a routing protocol to perform well with a wide range of network connectivity. When infrastructure is present, our proposed forwarding strategy helps improve the energy-delay trade-off when cost is a constraint.

DEDICATION

I dedicate this dissertation to my parents.

ACKNOWLEDGEMENTS

There have been many people who have walked alongside me in the past six years. They have guided me, supported me. Without them, I could not possibly finish this dissertation.

I would like to thank Dr. Radu Stoleru for his support, guidance, and patience. He encouraged me to bravely face tough challenges, to solve hard problems, to always pursue high standards. If I get stuck, his advice would guide me through. I could not complete this work without his help. I would also like to thank my committee members, Dr. Riccardo Bettati, Dr. Anxiao Jiang, and Dr. Alexander Sprintson, for their insightful comments. Their feedback and inputs to the research have shaped this dissertation.

I am grateful to be a member of a wonderful lab and to have worked with many talented colleagues. I would like to express my gratitude to Ala, Amin, Harsha, Jay, Mahima, Mengyuan, Myounggyu, and Wei, for their valuable inputs through insightful discussions.

I would like to thank my family: my parents, Fukuan Yang and Cuiying Hai, and my sister, Ning Yang. They kept me going. This work would not be possible without them.

Last but not least, I would like to thank my beautiful girlfriend, Zhubi You, for her love and endless support.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Dr. Radu Stoleru, Dr. Riccardo Bettati, and Dr. Anxiao Jiang of the Department of Computer Science and Engineering, and Dr. Alexander Sprintson of the Department of Electrical and Computer Engineering.

All work for the dissertation was completed by the student, under the advisement of Dr. Radu Stoleru of the Department of Computer Science and Engineering.

Funding Sources

Graduate study was supported by a scholarship from Texas A&M University.

This work was made possible in part by National Science Foundation (NSF) under Grant Number #1127449, #1145858, and National Institute of Standards and Technology (NIST) under Grant Number #70NANB17H190. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NSF and NIST.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Recent years have witnessed the steady increase of the number of mobile devices and the explosion of mobile data traffic. According to [2], such trend is expected to continue: comparing 2016 to 2021, the number of global mobile devices is expected to grow from 8.0 billion to 11.6 billion, while the monthly mobile traffic is expected to increase seven fold. People are increasingly relying on mobile devices (which range from wearables, smartphones, to tablets or laptops) to accomplish different tasks, to fetch and share information, or to conduct social interactions with family and friends.

The key enabling factor for applications in these mobile devices is networking, which currently heavily depends on a centralized infrastructure, i.e., cell towers, access points, and the core Internet in the back end. Consequently, most applications cannot work in their current form if the infrastructure is not available due to whatever reasons, e.g., device is out of coverage, infrastructure is destroyed by disasters, or is heavily congested. As an example, 89.3% of the cell sites are out of service even after nine days since a major hurricane hit Puerto Rico [14]. This prevents millions of people from getting access to critical information or staying in touch with families.

With the prevalence of mobile devices nowadays, we envision that Opportunistic Mobile Network (OMN) will play a critical role in the future mobile networks. An Opportunistic Mobile Network refers to the network paradigm where wireless devices communicate with each other through the opportunistically formed wireless links, and hence it does not depend on the infrastructure. In such networks, mobile devices are typically equipped with one or more wireless interfaces, such as WiFi or Bluetooth, in addition to the cellular interface. When two devices are within each other's communication range, a wireless link (using WiFi or Bluetooth) may form

and they can communicate. The formation of the wireless links heavily depends on nodes' mobility, which is assumed to be random and is captured by certain mobility models [11] [74]. In contrast to traditional networks, an end-to-end path between two devices may not always exist in OMN. Consequently, the data has to be "stored-and-forwarded" by mobile nodes: the node stores the data and forwards it to another node that is "closer" to the destination (measured by certain metrics) or to the destination itself. During this process, the data is often replicated multiple times [82] [79] before it reaches the destination so as to increase the delivery probability and reduce the delay. Compared to the traditional network where the delivery delay is usually at the millisecond scale, in OMN the delay may range from seconds to hours depending on the forwarding strategy being used, the mobility pattern, etc.

We argue that the design of a communication framework which contains efficient mechanisms and protocols that help route the data for OMNs is of utmost importance for future mobile networks. First, in many cases where the infrastructure is not available (e.g., disaster response scenarios, remote areas, etc.), such a framework provides best-effort communication services and has little deployment costs. Second, the ubiquitous existence of mobile devices nowadays creates more opportunities for leveraging opportunistic communications. Having such a framework is beneficial even under the coverage of infrastructure: for network providers (e.g., AT&T), it is able to offload a large amount of traffic originally carried by the already congested infrastructure [47] [31]; for end users, it provides an alternative way of communication, allowing them to trade off delivery performance with better energy efficiency and lower monetary cost [68].

Most previous work in OMN generally assumes that the network exhibits homogeneous connectivity characteristics. Some works [82] [79] [17] [4] [34] [27] [28] consider a purely infrastructure-less network with extremely sparse connectivity.

They assume that only pairwise contacts exist between mobile nodes. Some other work [68] [47] [26] [36] consider infrastructure but with only opportunistic contacts between mobile nodes and the infrastructure.

However, in contrast to the assumption made by existing works, we observe that users in OMNs actually experience *diverse connectivity*: the randomness of mobility and the large number of mobile devices lead to the dynamic formation of complex network topologies [19] [80] [92]; and the frequent interaction between mobile devices and the infrastructure leads to the simultaneous existence of mobile-to-mobile and mobile-to-infrastructure opportunistic contacts [52] [51]. Existing works which make incorrect assumptions often exhibit suboptimal behaviors. For OMN routing protocols [82] [4] [9] [48] [79] that are based on the pairwise contact assumption, redundant transmissions may happen when a node encounters multiple neighbors at the same time, whereas the routing protocol only makes forwarding decisions based on each individual link. Recent proposed social-based routing protocols [17] [34] [27] [92] [28] reduce data replication by exploiting high contact rate nodes, which, however, results in highly skewed energy consumption distribution. For data offloading protocols [47] [68] [26] that only consider opportunistic contacts between mobile nodes and infrastructure, mobile-to-mobile contacts, which can improve delivery performance without incurring too much monetary cost, are simply ignored. For the network that may dynamically evolve from sparsely connected to well connected (such as disaster response networks [13]), neither existing OMN routing protocols nor mesh network protocols, e.g., Optimized Link State Routing (OLSR) [37], work satisfactorily [80]. Yet, existing hybrid routing protocols [84] [18] [70] [61] [80] are either unrealistic in real world situations or perform poorly in dynamic scenarios.

1.1 Motivation

To overcome the limitations of existing solutions, we present the design of a communication framework for OMN with diverse connectivity. The proposed communication framework contains a set of routing protocols and algorithms. The upper level applications can choose to execute a subset of the protocols based on its own objectives. We approach the problem by moving along the connectivity spectrum, and solve problems with increasing complexity. On one end of the spectrum where the network connectivity is relatively sparse, mobile devices meet with each other opportunistically and form dynamic and unstable topologies. Since there are many existing routing protocols that make the assumption of pairwise contact, they perform inefficiently in such environments. Consequently, a scheduling algorithm is needed in order to improve the transmission efficiency of these protocols. On the other end of the spectrum where the network is well connected, existing routing protocols such as OLSR [38] are mature enough. However, a robust protocol that can adapt itself across the entire connectivity spectrum is still missing from the literature. Finally, in areas where infrastructure coverage (e.g., cellular network) is available, mobile devices may simultaneously have both opportunistic contacts and cellular connections. In this case, it is desirable to have a forwarding strategy that can achieve energy-delay trade-off while at the same time is cost-aware.

1.1.1 OMN With Sparse Connectivity

Traditional OMN routing protocols [82] [79] [48] [4] [9] assume a pairwise contact model: only two nodes are involved in an opportunistic contact at a time. In reality, however, even in a network with relatively sparse connectivity, multiple nodes may be in contact simultaneously. Through real world experiments using state-of-art implementation of OMN protocols, we find that the assumption of pairwise contact model

results in redundant transmissions when multiple nodes are in contact. To cope with this problem, we observe that: 1) nodes can take advantage of the broadcast nature of wireless communications; and 2) packet replications create tremendous opportunities for network coding in OMN. Motivated by these observations, we believe that a network coding based transmission scheduling algorithm can be used to solve the problem. It is, however, challenging to design such an algorithm as the problem is intractable even with simplifying assumptions.

Recent proposed social-based routing protocols [17] [34] [92] [28] [27] utilize social network analysis to identify "popular" nodes, i.e., nodes with high contact rates to other nodes, for efficient forwarding without massive packet replications. However, intuitively, such protocols may over-utilize the popular nodes. Through both theoretical and empirical analysis, we find that these protocols indeed result in severe energy consumption imbalance problem, where "popular" nodes consume much more energy compared to typical nodes. If we avoid the usage of "popular" nodes, however, more packet replications are needed to maintain the same routing performance. It is therefore challenging to develop a routing protocol which balances the energy consumption while at the same time maintains comparable routing performance without incurring too much overhead.

1.1.2 OMN Across Entire Connectivity Range

Real world wireless networks often exhibit diverse connectivity characteristics, instead of either sparsely connected or well connected. Some networks [19] have unreliable links among statically deployed nodes due to interference, unreliable power, etc. which leads to dynamic network topology and network partition. Other networks, such as ad hoc networks deployed in disaster scenarios [13], have highly mobile nodes (e.g., first responders, ambulances, etc), which leads to highly dynamic network

topology that ranges from well connected to disconnected. Yet, other networks [80] have both statically deployed mesh networks and mobile nodes in the same area, which leads to spatially diverse connectivity.

However, none of the existing routing protocols designed for OMN or mesh network deliver satisfactory performance [80] in such networks. Routing protocols for OMNs utilize packet replication for improving delivery performance, as contacts are assumed to be sparse. Such a mechanism incurs a significant overhead in well connected mesh networks. On the other hand, routing protocols for mesh networks, such as OLSR, adopt a single copy approach since an end-to-end path is assumed to always exist between any two hosts. However, in sparsely connected and highly dynamic mobile networks, such an approach fails to deliver satisfactory performance.

Moreover, existing hybrid protocols are either impractical or perform poorly. Some protocols [70] [18] [19] extend existing mesh routing protocols by adding store-and-forward mechanism so that data packets can reach disconnected nodes. This approach, however, performs poorly in highly dynamic mobile environments. Other protocols [61] [44] [84] [45] incorporate a OMN protocol and a mesh protocol, and then switch between the two protocols. However, the switching point can only be determined empirically and thus it fails to generalize to other scenarios. R3 [80] identifies that packet replication is the key structural difference between OMN and mesh routing protocols. However, the algorithm for dynamic replication has exponential time complexity when supporting more replications and the protocol adopts the source routing technique, which has been proven to be inefficient in dynamic environments [58].

1.1.3 OMN With Infrastructure Coverage

Previous OMN routing protocols consider a purely infrastructure-less network, whereas the prevalence of smart devices and the ubiquitous coverage of cellular network create an OMN that coexist with infrastructure-based network. Notice that cellular communication usually incurs non-negligible monetary cost [78] and higher communication energy consumption than WiFi or Bluetooth [32], but the data can be delivered instantly. On the other hand, protocols for OMN usually utilize WiFi or Bluetooth that incur negligible monetary cost and lower energy consumption, but only leveraging opportunistic contacts may also incur high delay. It is therefore desirable to design an integrated solution that leverages both opportunistic communication and infrastructure network to improve energy-delay tradeoff with monetary cost as a constraint.

Existing solutions [5] [68] [36] [26] [53] are inefficient in various ways. Some works [68] [36] ignore *opportunistic forwarding* among mobile devices, even if existing research [80] [89] has shown that leveraging high contact rates nodes may lead to higher offloading success rate, and hence lower energy consumption and monetary cost. Other work [53] [51] consider opportunistic forwarding among mobile devices. However, they do not explicitly consider energy consumption and monetary cost which can be important to the users.

It is, however, challenging to design a coherent offloading strategy that simultaneously considers energy consumption, delay and monetary cost while leveraging opportunistic forwarding. Specifically, opportunistic forwarding may incur more energy consumption if replication is used since more transmissions are involved. On the other hand, opportunistic forwarding with replication may lead to higher offloading success rate which may, in turn, reduce both energy consumption and monetary cost.

1.2 Dissertation Statement

As Opportunistic Mobile Networks often exhibit diverse connectivity, i.e., the network may range from sparsely connected to well connected and may coexist with infrastructure, the design of a communication framework, which consists of a transmission scheduling algorithm (for optimizing transmissions over multiple links) and a set of efficient, problem specific routing and data forwarding protocols, each of which optimizes for certain objectives and is designed with diverse connectivity in mind, is critical in order to provide energy efficient, robust and cost-aware communication service to users in Opportunistic Mobile Networks.

1.3 Main Contribution

The main contributions of this dissertation are listed as follows:

- We propose the **Multi-node Contact Optimization Algorithm** (MCO) [86]. It is routing protocol independent and thus can work with any OMN routing protocol that is replication based. The algorithm receives the routing decisions as input, and outputs a transmission schedule that consists of a series of node-packet pairs that specifies the transmission order. The packets specified in the transmission schedule are either unaltered packets or network coded packets. The objective is to improve the *energy efficiency* through the reduction of the number of redundant transmissions.

- We propose the **Energy Balanced Routing Protocol** (EBR) [87]. It balances the energy consumption among different nodes during the routing process, while at the same time maintains the routing performance. We present the design of a novel social-based routing metric, a protocol independent intra-connected-component routing mechanism, as well as a theoretical performance

analysis of the proposed protocol.

- We propose the **Hybrid Routing Protocol** (HRP) [89]. It is designed to achieve *robust* performance in wireless networks with a wide range of connectivity characteristics, i.e., ranging from well connected wireless mesh networks to sparsely connected OMNs. We present a novel mathematical model that is able to capture not only the inter-contact times but also the correlation information, which have significant impact on the benefit of data replication. We also present a novel regret-minimization based algorithm to dynamically decide the amount of replication. We show how HRP can be implemented on real hardware.

- We propose the **Cost-aware Energy-efficient Data Offloading Protocol** (CEO). It forwards data packets from mobile nodes to a remote server while minimizing the energy consumption with the constraint of deadline and monetary cost. It leverages both opportunistic mobile-to-mobile and mobile-to-infrastructure contacts. We show that the problem can be formulated as a discrete time optimal control problem, and that the optimal policy has a simple threshold-based structure. We present an approximation based algorithm that can calculate the policy efficiently.

- We evaluate the proposed algorithm and protocols [86] [87] [89] through both simulations and real world experiments. For simulations, we adopt the Opportunistic Network Environment (ONE) [42] simulator, and we use both synthetic mobility models (e.g., Truncated Levy Walk [74]) and a wide range of real world mobility traces [22] [55] [77] [73]. For real world experiments, we implement a proof-of-concept prototype of HRP and deploy it on wireless routers. We

emulate a large range of network conditions to thoroughly evaluate the performance.

1.4 Organization

This dissertation is organized as follows. In Section 2, the state-of-art solutions are presented. In Section 3, we present the overall system architecture of the proposed communication framework. In Section 4, the design of the Multi-node Contact Optimization algorithm is presented. In Section 5, we present the Energy-balanced Routing protocol. In Section 6, we present the design and implementation of the Hybrid Routing Protocol. In Section 7, the Cost-aware Energy-efficient data offloading protocol is presented. Finally, in Section 8, we conclude this dissertation and present a few future directions.

2. STATE OF THE ART

In this section we present the state of the art of this dissertation. We first present the related works for OMN routing protocols, which includes traditional replication based protocols, social network analysis based protocols, as well as protocols that attempt to balance the energy consumption. Next, we discuss the hybrid routing protocols that are designed for networks with a wide range of connectivity characteristics. Finally, we present the related work about data offloading problem where opportunistic communications and infrastructure are leveraged simultaneously.

2.1 Related Work for OMN Routing Protocols

2.1.1 Traditional OMN Routing Protocols

Routing protocols for OMNs have been extensively studied [82] [48] [79] [9] [4]. Epidemic [82] proposed that by exploiting pairwise contact processes, it is possible to route packets in intermittently connected networks. During each contact, the node requires whatever packets it does not possess from the encountered neighbor. This flooding-based routing protocol achieves optimal delay and delivery ratio performance when given unlimited bandwidth and unlimited storage at each node, but with the cost of large overhead. Epidemic is often used as upper bound for performance evaluation. In Spray&Wait [79], the source "sprays" a limited number of messages into the network; these message carriers then simply wait for the destination to deliver the message. In this way, it initially "jump-starts" spreading the messages into the network like what Epidemic does. When enough copies of the packet are injected into the network, it stops data replication and let each node carrying a copy perform direct transmission.

Flooding-based (or restricted flooding-based) protocols require no knowledge of

the network in advance. However, it is shown in [39] that the increased knowledge about the network would improve routing performance in OMN. As such, protocols [48] [9] [4] try to predict future mobility pattern by using contact history. Prophet [48] makes routing decisions by calculating the probability for the encountered node to deliver a packet to its destination. The basic idea is to maintain a delivery predictability for each pair of the nodes. This predictability is updated each time these nodes meet, and is also aged as time elapse when they are not in contact. Then during a contact, the messages are forwarded to the node with higher delivery probability. In [9], delivery probability is estimated by contact frequency. Then the packets with high delivery probability and the packets that have not traversed far in the network are given higher priority. A more advanced buffer management mechanism is also proposed in order to prioritize the packets to be dropped. RAPID [4] views routing as a resource allocation problem. In order to optimize a certain routing metric, such as minimizing average delay of the packets, they derive a per-packet utility function and make the routing decision by maximizing the marginal utility. [59] proposes encounter-based routing. The key idea is also to keep track of the encounter rate of a node, i.e., the number of times it encounters a neighbor. When two nodes are in contact, the relative ratio of their encounter rate determines the proportion of message replicas to exchange.

Notice that all these routing protocols are based on the pairwise information exchanging in order to make routing decisions. When the node encounters a neighbor, it checks each message within its buffer and make a routing decision of whether to forward this message to the neighbor. Nodes are thus unaware of the potential multiple ongoing links.

2.1.2 Social-based OMN Routing Protocols

Previous OMN routing protocols are usually built using observed mobility patterns. The dynamic and unpredictable topological changes make the routing structure detected in this way unreliable. In many cases where the mobile devices are human-held devices, social characteristics such as friendship, community structures, etc. are less volatile than mobility [34].

SimBet [17] is a single-copy routing protocol based on social network analysis. The key idea is to identify "bridge" nodes in order to facilitate the routing process. In particular, the *betweenness* and *similarity* are used by nodes to decide whether to forward a message to the encounter node. The author propose to use "ego network analysis", which can be performed locally by individual node using only local information, to estimate such metrics. The routing metric *SimBet* then consists of a weighted summation of *betweenness* and *similarity*.

BUBBLE RAP [34] is one of the most famous social-based forwarding strategies. The key idea of BUBBLE RAP is to forward the message towards more popular nodes and towards the community in which the destination belongs. Once the message is forwarded to the destination's community, it can only be forwarded to more popular nodes within the same community, until it is delivered to the destination. The popularity is measured by degree centrality, i.e., the number of unique neighbors a node has met during a time slot.

In [29], a new routing metric called Cumulative Contact Probability (CCP) is proposed as an alternative metric for "betweenness". The authors show that there is a better correspondence between CCP and the chance for a node to lie on the shortest path among other nodes than betweenness metric.

Community-aware Opportunistic Routing (CAOR) [85] is a routing protocol

based on social communities. Instead of routing among mobile nodes, CAOR intends to route data among communities. The paper assumes that for each node, there are a few locations, called *home*, that the node frequently visits. In each home, there is a *throwbox* which can storage messages. The the social network analysis is used to derive expected delays between homes, and the forwarding strategy for a node is simply to forward to home which has smaller expected delay to the destination's home.

In addition to social community, the temporal information is also shown to be important in the process of information dissemination [66]. The key contribution of [66] is the methodology of breaking contact traces into *temporal communities*, where multiple nodes meet each other frequently and for a long period of time. The authors show that these temporal communities have high correlation with the actual social communities obtained through analysis of friendship, share of affiliation, home city, etc. Moreover, the detected temporal communities are also used for analysis of information dissemination in opportunistic social networks. It turns out that high contact rate nodes are responsible for most information dissemination. Particularly, nodes that are only frequently active within temporal communities have less impact on information dissemination. The high contact rate nodes which connect different communities play a more important role in the information dissemination process.

[91] also identify the importance of temporal community, which they termed as *transient community*. In addition to the analysis of transient community, they developed a forwarding strategy. The basic idea is to keep track of the node's current transient community, and use this information to decide whether the encountered node has higher probability to meet a node which is in the community where the destination belongs to. Packet replication is used to decrease the delay.

In [49], the authors analyze a mobility dataset of roughly 200 freshmen students

of University of Notre Dame over 15 months, in order to demonstrate the feasibility of opportunistic communication and explore the potential of using opportunistic communications to augment the existing infrastructure based wireless network. The results show that opportunistic communication opportunities are prevalent, that the opportunistic contacts are stable (contacts maintain long duration, and have high frequency), and that opportunistic contacts have fairly good reciprocity (nodes can achieve a balance between receiving service and offering service).

[92] identifies the existence of *Transient Connected Components* (TCC) through empirical analysis over five mobility traces, and the authors prove that if the TCCs satisfies certain properties, the contact opportunities between mobile nodes are increased. They propose data forwarding strategies that are aware of the existence of TCC, where each node is associated with a *centrality* metric. In the naive strategy, the data is forwarded to the node with highest centrality in the TCC. In the enhanced strategy, the data replicas in the TCC are rearranged based on a *set centrality* metric, which characterizes the total delivery capability of the nodes currently holding the data. However, the forwarding decisions are made centrally at a command node, which is less scalable in large networks.

2.1.3 Energy-balanced OMN Routing Protocols

A common pattern for the social based protocols is that the data is forwarded to more "popular" nodes as they are more capable of delivering data. However, this may lead to energy consumption imbalance problem. Yet, the energy consumption imbalance problem was thoroughly explored in the literature. In [23], the authors analyze the imbalance problem for delegation forwarding, i.e., a forwarding strategy where the node only forwards to another node with the highest metric it has ever seen, without the consideration of TCC, but provide no solution to the problem.

FairRoute [67] proposes to use queue length to limit the traffic received by popular nodes. A node would forward a message only if the encountered node has higher quality metric and smaller queue length. Similarly, in [46] the authors use buffer space advertisements to avoid storage congestion. In [69], a congestion control framework is proposed to offload traffic from congested individual nodes or network parts to other parts of the network. The framework mainly consists of a set of utilities and an adaptive replication rate. Although congestion control mechanism has positive impact on traffic balance, it does not necessarily guarantee that the aggregate traffic, which is directly related to energy consumption, is also balanced among nodes. Popular nodes are able to deliver messages quickly to destinations, and those messages are therefore purged from their buffer, which makes room for the new messages to be received and the aggregate traffic may still be high. These protocols also do not consider TCC and ignore the fact that popular nodes may also have higher chance to relay messages for other nodes within a TCC. Our proposed solution tries to directly balance the aggregate traffic and avoid hot-zones within TCCs.

2.2 Related Work for Hybrid Routing Protocols

Previous works [19] [70] [18] [44] [84] [80] [61] [45] [51] have investigated the integration of routing protocols for connected networks and sparsely connected networks using different approaches.

One of the most significant difference between routing in connected networks, i.e., mobile adhoc networks (MANET) or mesh networks, and in OMNs is that OMN protocols typically adopt the store-and-forward mechanism in order to deliver the packets. Hence, a naive approach of integration is then to embed the store-and-forward mechanism to a MANET/Mesh routing protocol [70] [18] [19], such that the data can be delivered even if the nodes are disconnected for some period of time. The

basic idea is to buffer the data whenever the next hop is unavailable, and retransmit the data when the next hop is available again or a new route is discovered. Although this may work well under specified network conditions [19] [70], the lack of ability for selecting appropriate packet carriers in disconnected networks and the limitation of using single copy limit its performance.

Another intuitive approach of hybrid routing is to switch between existing MANET and OMN protocols [61] [44] [45] [84] [51]. The source decides, based on certain conditions, whether to use MANET or OMN routing protocols. For example, in [45], the source decides which protocol to use based on its local information about the node density, message size and path length to the destination. If the network is sparse, or the nodes are moving fast, or the message size is large, then OMN routing is used; otherwise the MANET routing is used. In [84], a different switching scheme is proposed. The key idea is to partition temporally connected network into multiple diameter-constrained groups, then MANET routing is used for intra-group routing while OMN routing is used for inter-group routing. The major problems for these solutions are as follows. First, it is hard to clearly define the boundary between OMNs and connected networks, which makes it hard to choose an appropriate switching point, e.g., the group diameter in [84]. Second, the switching decisions are made at the source, and hence the intermediate nodes cannot change the decision even if they may have more up-to-date information. Third, even if the protocol allows the intermediate nodes to make additional decisions, as have been proposed as an alternative in [80], it may change between different protocols frequently which results in sub-optimal routes and it still makes poor decisions in terms of data replication which results in longer queue at each node [80].

An important observation was made in R3 [80] that replication is the key structural difference between routing protocols for well connected networks and OMNs.

They further discovered the relationship between replication benefit and the path delay predictability. Similar observation of the relationship between replication benefit and link delay predictability is also made in data center networks [83]. Utilizing their analytical model for replication, R3 is able to dynamically replicate packets according to the network condition. To do this, each node collects path delay distributions by calculating the convolutions of individual link delay distributions obtained through probing the network. The shortest path in terms of expected path delay is first selected. Additional paths are used if the calculated replication gain exceeds a certain threshold. The source then puts the path(s) in the packet header for source routing. Unfortunately, the computational cost grows exponentially when trying to support more replications and the adopted source routing technique was shown to be inefficient in highly dynamic OMN environments [40].

Inspired by R3, our work delves deeper into understanding replication. We find that *the correlation of path delays can significantly reduce the benefit gained from packet replication*. We propose a novel model for capturing the potential correlation of inter-contact times, which is the major source of packet delay in highly dynamic OMN environments, among a group of nodes. We further propose a novel adaptive algorithm for dynamically and efficiently choosing the number of packet replications, which take the *delay correlations* into account for better decisions.

The impact of correlated mobility on forwarding efficiency is explored in [8]. Our work differs from their protocol in the sense that we also study the impact of correlation on replication and that we propose a new model for better capturing correlation. Through simulation, we show that our protocol achieves superior performance.

2.3 Related Work for Data Offloading Through Opportunistic Communication

Mobile data offloading has been an active research area. Among many other works, user side data offloading is the most relevant to our work. Ra et al. [68] proposed SALSA, which leverages Lyapunov optimization framework to minimize the energy consumption while maintaining the queue stability. Mehmeti et al. [56] analyzed the delay-energy tradeoff of delayed offloading. Their work focus on the case where the message is forwarded at the first contact with an AP, while our work extend the idea and allow further delaying if the transfer opportunity is energy inefficient. Gao et al. [26] proposed online and offline algorithms to schedule data transfer when AP's capacity is limited. Similarly, [36] schedules data transfer based on predicted WiFi availability, future traffic and user preference. Different from our work, these work do not consider opportunistic forwarding, while [26] and [36] do not consider energy consumption either. Lu et al. [53] proposed cooperative data offloading, where the forwarding decision is made based on opportunistic paths' availability and delivery probability. Different from our work, they do not consider the energy consumption of data transfer. For a comprehensive survey of mobile data offloading, readers are referred to [71].

Existing works [6] [35] that study the impact of infrastructure on opportunistic communications, or vice versa, mainly focus on delay and PDR. Different from their work, we try to explore the trade off between energy consumption, monetary cost and delivery performance when both opportunistic and infrastructure communication are considered.

Khouzani et al. [43] uses optimal control framework to design energy efficient Epidemic routing protocol. However, their work considers a homogeneous network, while ours consider a more complicated heterogeneous network. Moreover, they

use the number of transmissions as an indicator of energy consumption, while we explicitly model the energy consumption for each transmission. Lastly, they do not consider monetary cost, while CEO adapts to users cost preference.

3. SYSTEM ARCHITECTURE

In this section, we present the overall system architecture of the proposed communication framework for OMNs with diverse connectivity. Before we delve deep into the technical detail, we first introduce a few general but important concepts presented throughout this dissertation.

3.1 Preliminaries

Opportunistic Mobile Networks fall into the general category of Delay Tolerant Networks (DTN) [24], which was initially designed for interplanetary networking. OMN can thus inherit the architecture designed for DTN. An application data unit in OMN is referred as a *message* in this dissertation, and a message may have arbitrary length [10]. To cope with high delay and potential frequent disconnection, nodes in OMN employ persistent storage for storing messages. All messages in OMN are also associated with a lifetime (or maybe called deadline in the rest of this dissertation), which indicates the time span when this message is useful. A message is dropped if it reaches its deadline. We refer to a node as a *message carrier* for a message if it carries a copy of the message. A node is the *next carrier* if the OMN routing protocol determines that a copy of the message should be transmitted to that node. Notice that this is different from the *next hop* as in traditional term, since the *next carrier* might be multiple hops away in a temporally connected network in OMN as we will see in later sections.

When two nodes encounter, they first exchange information that facilitates routing, which may include a summary vector of the carried messages, routing protocol dependent meta data, etc. Then for each of the messages in its storage, a routing decision has to be made. In general, deliverable messages, i.e., the ones whose des-

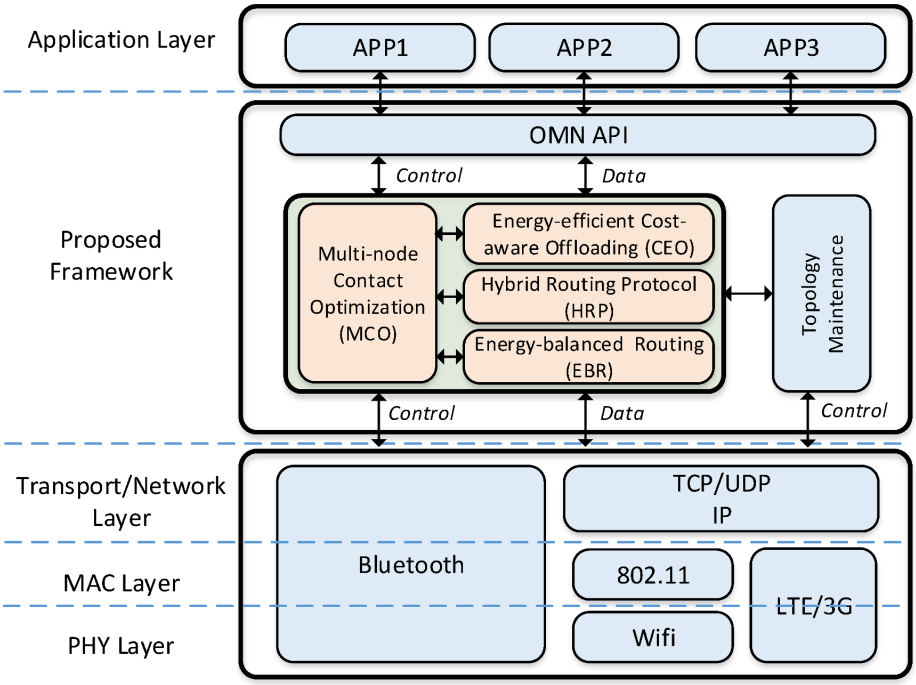


Figure 3.1: Overall system architecture

tination is the encountered node, are sent first, followed by other messages. If the routing protocol is replication based, then the sender will keep a copy of the message; otherwise it drops the transmitted message.

3.2 Communication Framework Architecture

In Figure 3.1, we present the overall architecture of the proposed communication framework. The proposed framework serves as the middleware which sits above the transport layer of the network stack, and provides energy efficient, robust and cost-aware communication services to the OMN applications. Mobile devices which utilize our framework form an overlay network above the network layer. Once the framework has made a routing decision, it simply utilizes the underlying protocols to transmit the data to the *next carrier*.

In order to leverage opportunistic communications, applications utilize the pro-

vided Application Programming Interfaces (API) to send and receive data instead of using the traditional sockets. The framework then takes custody of the data and delivers the data in its best effort while optimizing for certain objectives. The framework leverages all communication interfaces, such as Bluetooth, WiFi, and cellular (e.g., LTE or 3G). Within the framework, we assume that there exists a Topology Maintenance module, which is able to provide the overall graph view of the (potentially multi-hop) network the node is currently connected to.

At the core of the framework is the set of three routing and data forwarding protocols (EBR, HRP and CEO) and a scheduling algorithm (MCO), each of which optimizes for certain objectives. The MCO algorithm takes input from the routing protocol and schedules actual transmissions in order to achieve energy efficiency, which can be leveraged by the three routing and forwarding protocols. The EBR protocol utilizes a new routing metric to conduct efficient OMN routing while maintaining energy consumption balance. The HRP achieves robust routing performance across a wide range of connectivity characteristics. The CEO utilizes both cellular interface and opportunistic links (formed by WiFi or Bluetooth) to achieve cost-aware energy efficiency data offloading. When activated, the protocol module fetches topology information from the Topology Maintenance module, exchanges control information with other nodes, and handles actual data forwarding. As each of the protocol optimizes different objectives, it is up to the application to choose which protocol to execute.

One potential implementation of the proposed framework is to integrate with existing DTN implementations, e.g., IBR-DTN [21]. The framework can be implemented as an independent component, which leverages the IBR-DTN library for services such as storage management, API, etc. and provides the necessary routing functionality. In our proof-of-concept implementation of HRP, we also leverages

olsrd [81], i.e., an implementation of OLSR, for topology management. This dissertation focuses on the theoretical aspects of OMN communications and proof-of-concept implementations. We leave the full-fledged implementation as future work.

4. MULTI-NODE CONTACT OPTIMIZATION ALGORITHM [1]

Opportunistic mobile networks are intermittently connected ad hoc networks, where information is disseminated through opportunistic wireless communication among nodes, whenever there is a contact [64]. This infrastructureless networking, enabled by human-held mobile devices [33], has many promising applications including disaster rescue [13] and mobile crowdsensing [25]. In these networks, an important research problem is routing. Recently proposed routing protocols use social network analysis to make routing decisions since humans' social structure is not prone to rapid changes [76]. Messages are forwarded from source to destination in a hop-by-hop manner. Any node can serve as a forwarder as long as it is able to bring the message *closer* to destination.

Although recent studies have demonstrated the practical potential for opportunistic communication using smartphones [49] and opportunistic mobile networks have been shown to support new paradigms of computing [57], several research challenges have not been satisfactorily addressed. For example, opportunistic forwarding has only been treated as a point-to-point contact between nodes, as opposed to a simultaneous contact among multiple nodes. This is possibly due to the fact that opportunistic mobile networks have been essentially treated as traditional delay tolerant networks (DTN), and some assumptions for DTNs were not re-considered for social networks. In DTNs, the contacts are assumed to be very sparse. Only pairwise contacts are considered when designing forwarding strategies [82], and thus, links are only responsible for transferring the message to the encountered node. Since con-

[1]Reprinted from "Routing protocol-independent Contact Optimization for opportunistic social networks" by Chen Yang, Radu Stoleru, 2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Larnaca, 2014, pp. 534-541. Copyright 2014 by IEEE.

tacts in opportunistic mobile networks may be more frequent than in traditional DTN, some nodes may have more than one neighbor occasionally. This has been confirmed recently in studies on temporal communities [91] [66], where the nodes in a group tend to meet each other frequently during some time window. Nodes with two or more neighbors may then take advantage of the broadcast nature of wireless link, instead of viewing the contact as multiple individual point-to-point links. Furthermore, the routing decision may also be impacted in situations when a node has multiple neighbors at the same time. For example, when multiple nodes are eligible to be the next hop for a packet, the current message holder may want to forward it to the next hop with the largest delivery probability, which requires the routing protocol be aware of multiple ongoing links.

In this section, we demonstrate that the pairwise contact model might not be appropriate for opportunistic mobile networks. We analyze both real and synthetic mobility traces to study how frequent and stable are contacts involving multiple nodes, i.e., multiple nodes temporally form a connected network. We will demonstrate that nodes have a high probability to be involved in multiple node contacts and that the temporally formed topology is fairly stable with respect to duration. These findings serve as our motivation to investigate how these multiple node contacts may be utilized to reduce the number of transmissions in opportunistic mobile networks.

We do not propose a new routing protocol in this section; instead, we develop a centralized network coding based contact optimization algorithm, which receives routing decisions as input and schedules packet transmissions such that the number of transmission is minimized for exchanging the same amount of information. The combination of network coding and wireless broadcast has demonstrated its capabilities of increasing the throughput in wireless mesh networks [41], but has not yet

been investigated in opportunistic mobile networks. We show through simulations that our algorithm is able to reduce the total number of transmission without impacting routing performance, i.e., packet delivery rate (PDR) and packet delivery delay(PDD).

This section is organized as follows. We present our analysis of multiple node contacts and inefficiencies for existing protocols in Section 4.1. The network model and the design of the network coding based Contact Optimization algorithm are introduced in Section 4.2. We evaluate our algorithm and analyze our simulation results in Section 4.3.

4.1 Multi-node Contact Analysis

We start from the key observation that in opportunistic mobile networks, in addition to pairwise contacts (i.e., contacts between two nodes only), a node has non-negligible opportunity to be involved in multiple node contacts, which we call *multi-node contacts*. Existing routing protocols which only treat contacts as point-to-point links may consequently have inefficient behaviors. In this section, we analyze the *frequency* and *stability* of the *multi-node contact* using realistic and synthetic mobility traces. We then show the inefficiencies of routing protocols that adopt the pairwise contact model.

4.1.1 Multi-node Contacts

4.1.1.1 Methodology

We study the frequency and stability of *multi-node contacts* by analyzing mobility traces. A mobility trace consists of a series of contact events, where each contact event contains two node's ID, and the starting and ending times of the contact. For ease of discussion, we first introduce several concepts:

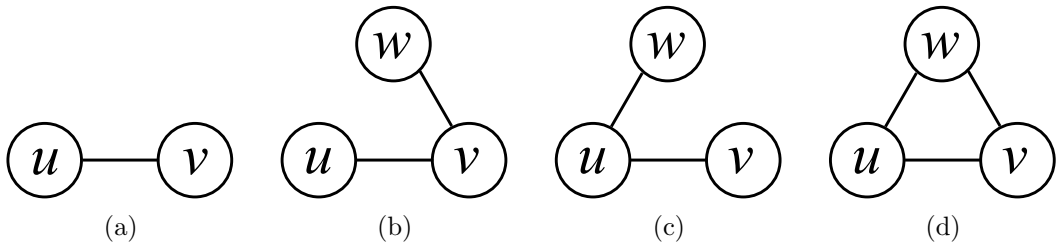


Figure 4.1: Multi-link illustration: (a) a pair-link between u and v; (b) (u, v) and (v, w) are labeled as multi-links; (c) (u, v) and (u, w) are labeled as multi-links; (d) all pair-links are labeled as multi-links. (Here we omit starting and ending time for simplicity.)

Definition 1. *Pair-link l: a pair-link l between nodes u and v is formed when they are in contact and disappears when the contact ends. It is denoted by a 4-tuple: $l = (u, v, t_s, t_e)$, where t_s and t_e are the starting time and ending time of the link. We denote the starting time of a pair-link l by $t_s(l)$ and the ending time of the link by $t_e(l)$.*

Definition 2. *Multi-link: we label a pair-link l as multi-link if during the time interval $[t_s(l), t_e(l))$, there exists at least another node w that is within communication range with either u or v, or with both.*

As an example, consider a pair-link l between u, v as shown in Figure 4.1a. If during $[t_s(l), t_e(l))$ another node w forms pair-links with u or v as shown in Figures 4.1b, 4.1c, or 4.1d, then l is labeled a multi-link. In this case, we say that u, v and w form a *multi-node contact*.

Definition 3. *Multi-link percentage p: For a network with multi-node contacts, $p = \frac{\#multi-link}{\#pair-link}$.*

The multi-link percentage reflects how *frequent* a pair-link is labeled as multi-link, and thus represents the *frequency* of of a multi-node contact. If a pair-link is labeled

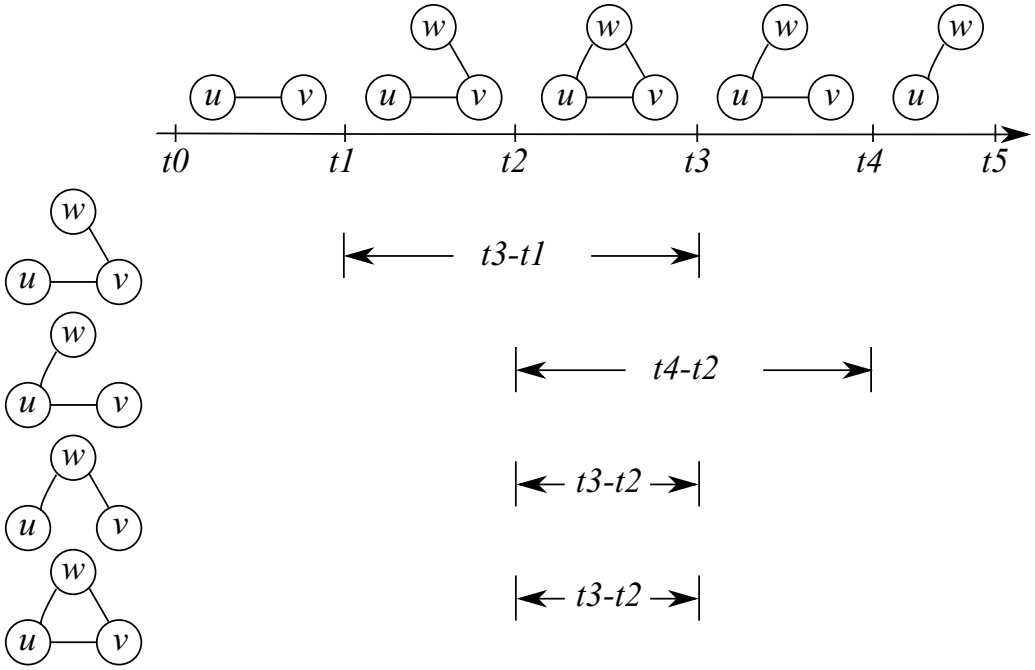


Figure 4.2: Example of a multi-node contact.

as multi-link, then at least one of its endpoints has the opportunity to maintain at least two pair-links at the same time. Notice that p is closer to 1 if nodes meet each other in a multi-node contact manner, instead of a purely pairwise manner. The value of $1 - p$ represents the fraction of links that are only pair-links.

In order to understand the *stability* of multi-node contacts, we calculate how long a connected topology (with more than two nodes) lasts until any of the pair-links ends. A connected topology is represented by a graph $G = (V, E)$, where V ($|V| > 2$) is a set of nodes and E is the set of pair-links connecting nodes. Notice that all pair-links in set E are also labeled as multi-links by definition. We define two types of durations as follows:

Definition 4. *(a) Topology duration: the duration of a topology, represented by* $G = (V, E)$*, is given by* $\min\{t_e(l), l \in E\} - \max\{t_s(l), l \in E\}$*; (b) Pair-link duration:*

the duration of a pair-link l is $t_e(l) - t_s(l)$.

We show an example of a three-node contact in Figure 4.2. In this example, nodes u, v and w form a multi-node contact. Pair-links among them are formed and end at different times. For this multi-node contact, there are four connected topologies total (as shown in the left hand side of the figure). For each connected topology, we also show its duration. For example, the clique topology formed at $t2$ lasts $t3 - t2$ until the pair-link v and w ends at $t3$.

4.1.1.2 Datasets and Results

We first analyze the possibility of multi-node contacts in real world mobility traces. For this, we use the following datasets: *Intel, Cambridge, Infocom'05, Infocom'06* [11]. These traces record contacts among users carrying Bluetooth devices. *Intel* includes eight researchers working at Intel Research in Cambridge, UK. *Cambridge* include twelve students and faculty, members of a research group at University of Cambridge. The last two traces are from 41 and 78 users, attendees at Infocom conferences. All traces are preprocessed and only contacts among users in the datasets are considered. We assume that the contacts are bidirectional.

Table 4.1: Multi-link percentage results for real world mobility datasets

	Intel	Cambridge	Infocom2005	Infocom2006
Mobile nodes	8	12	41	78
Multi-link percentage(%)	86.1	76.7	94.2	99.1
Pair-link duration(s)	1,112	618	492	343
Inter-contact duration(s)	5,443	3,238	10,052	5,572

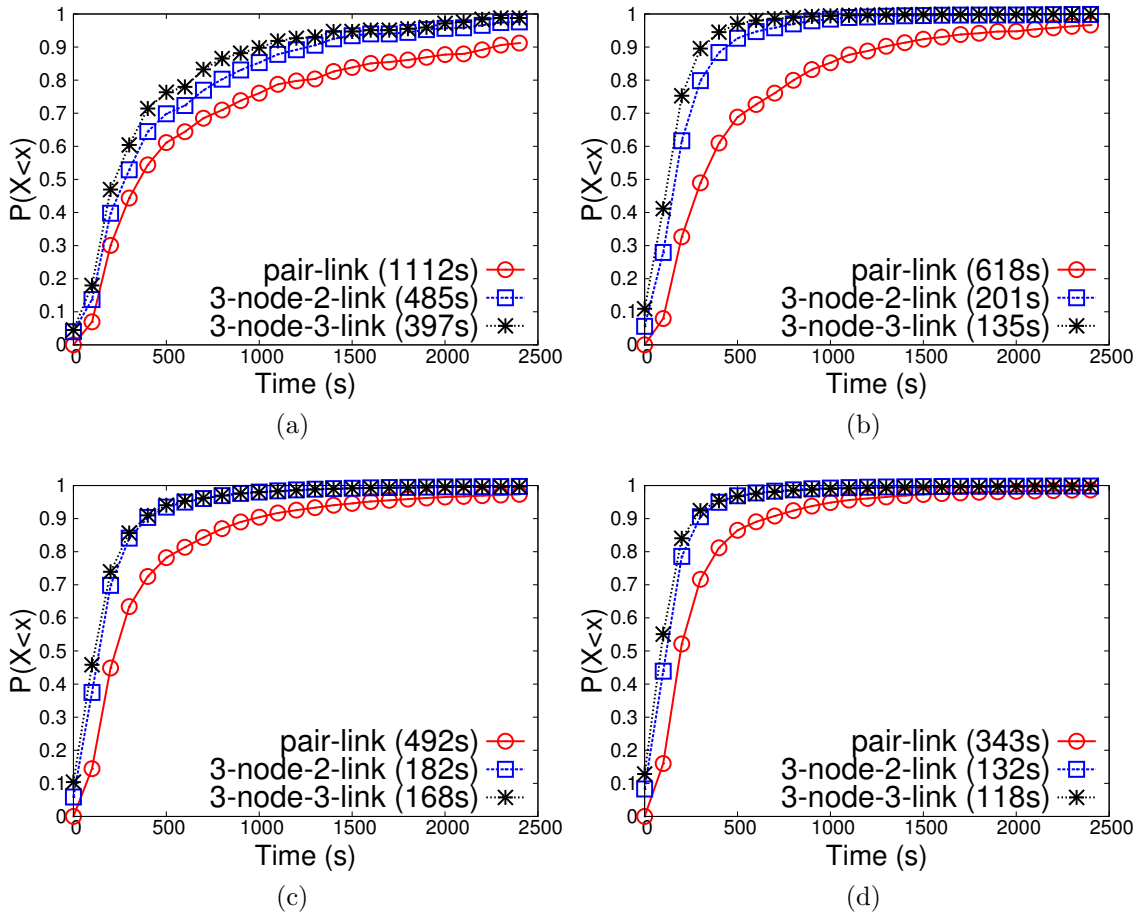


Figure 4.3: Link duration CDF for real world trace: (a) Intel; (b) Cambridge; (c) Infocom2005; and (d) Infocom2006.

We present the multi-link percentage results in Table 4.1 and duration results in Figure 4.3. We found that the majority of pair-links formed in these mobility traces can be labeled as multi-links, since the ratio between the number of multi-links and pair-links is close to 1. This indicates that only a few number of links are pair-links. Since multi-links imply multi-node contacts, we conclude that multi-node contacts occur frequently in these mobility traces.

Since the number of possible topologies increases exponentially with the number of nodes involved in a multi-node contact, here we only show the CDF of durations for

the topologies presented in Figures 4.1b through 4.1d. We denote topologies similar to Figures 4.1b and 4.1c as "3-node-2-link" and the topology in Figure 4.1d as "3-node-3-link". We also include the duration of pair-links as baseline for comparison.

We plot the duration CDF for each topology and pair-link in Figure 4.3. We also show the mean value in parentheses. From the results shown in Figure 4.3, we can observe non-negligible durations for linear and clique topologies. Since the duration of these three-node topologies is relatively long, a non-negligible number of packets can be exchanged among nodes. We conclude that multi-node contacts in these simple topologies are stable enough and can be utilized to increase transmission efficiency.

4.1.1.3 Impact of Node Density and Communication Range

Since real world mobility traces do not allow us to examine the impact of node density and communication range over multi-node contact, we use a Truncated Levy Walk (TLW) mobility model [74] to generate a series of synthetic traces. TLW is a random walk model that emulates statistical features of human movement such as the heavy-tail distribution of flight length and the pause time. At each step, a node chooses a direction uniformly at random. It then chooses the flight length and pause time randomly from Levy distributions with coefficients α and β, respectively, and with truncation factor τ_l and τ_p. We use $\alpha = \beta = 1.8$, $\tau_l = 200m$ and $\tau_p = 1h$, as it was used to simulate *Infocom'05* [74]. Other simulation variables are listed in Table 4.2.

Figure 4.4 shows the results from synthetic traces. Figure 4.4a shows that the percentage of multi-link increases almost linearly with the communication range. It also increases with the number of nodes in the area. It can be seen from Figure 4.4a that in sparse networks, multi-node contacts may still occur. When there are 40

Table 4.2: Parameters of synthetic mobility trace

Number of nodes	40, 50, 60, 70, 80
Communication range(m)	50, 60, 70, 80, 90, 100
Area(m^2)	1500 × 1500
Simulation time(hr)	24

nodes in the area and the communication range is 100m, the average node degree is roughly 0.56. Even in this scenario, more than 70% of the pair-links are multi-links.

Figures 4.4c and 4.4d show the average duration of different topologies. We note that communication range is the key factor that influences the duration for all topologies. More importantly, the durations of "3-node-2-link" and "3-node-3-link" are almost half of the pair-link duration, as shown in Figure 4.4b. This further validates that multi-node contacts in these topologies are stable.

4.1.2 Inefficiencies of Existing Routing Protocols

Now that we have validated the existence of multi-node contacts in opportunistic mobile networks, we examine *what might be the inefficiencies if routing protocols view links only as pair-links*. We conduct experiments on real wireless mesh hardware, illustrating redundant transmissions for protocols adopting the replication mechanism, as this has been widely used for decreasing delay in opportunistic networks.

In our experiments we used three Mikrotik RB433UAH routers [1] (labeled u, v, and w) running IBR-DTN [21] on top of the OpenWRT operating system and using Epidemic as routing protocol. The bundle size was set to 100kB. In the first experiment, we let w send 20 bundles to a non-existing node d. Then we turned on nodes u and v simultaneously, forming a multi-node contact as shown in Figure 4.1d. We collected node w's sending times for each bundle and plot them in Figure 4.5a. In the second experiment, we let u send 20 bundles to a non-existing node d. Then

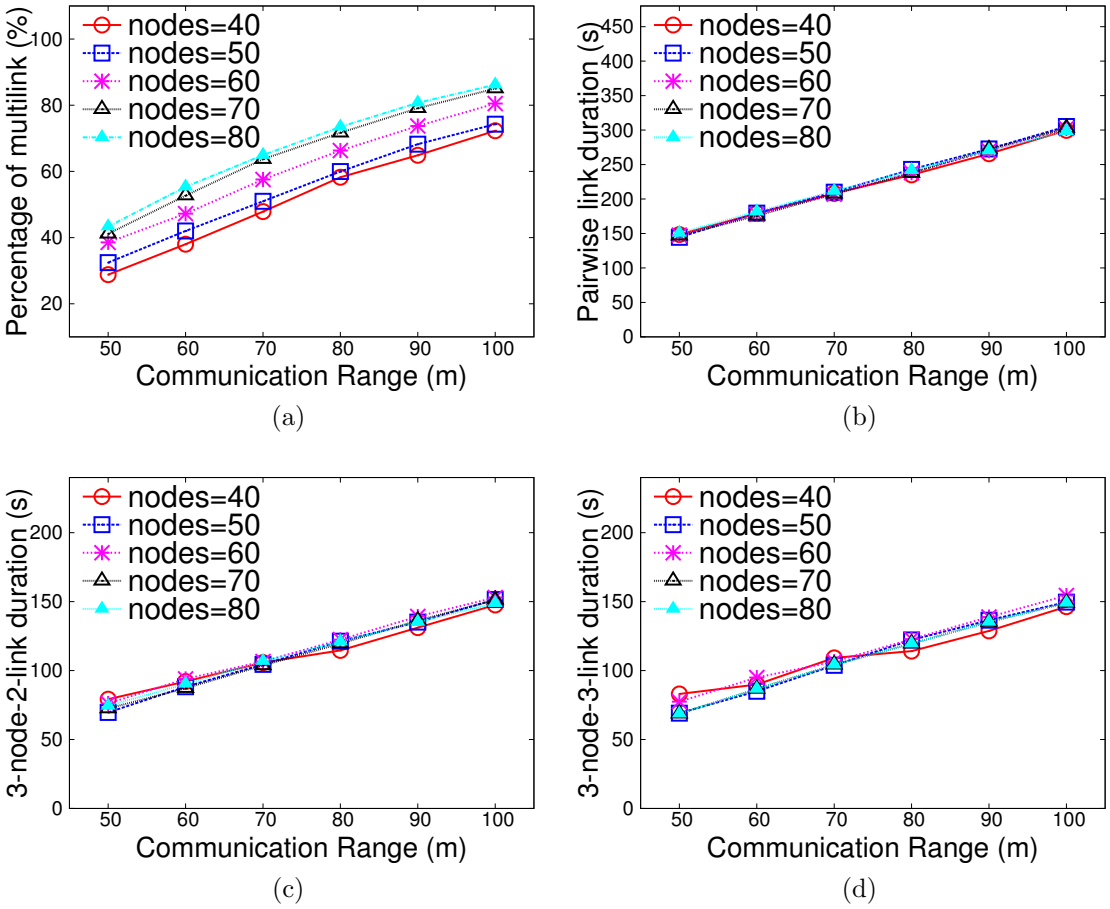
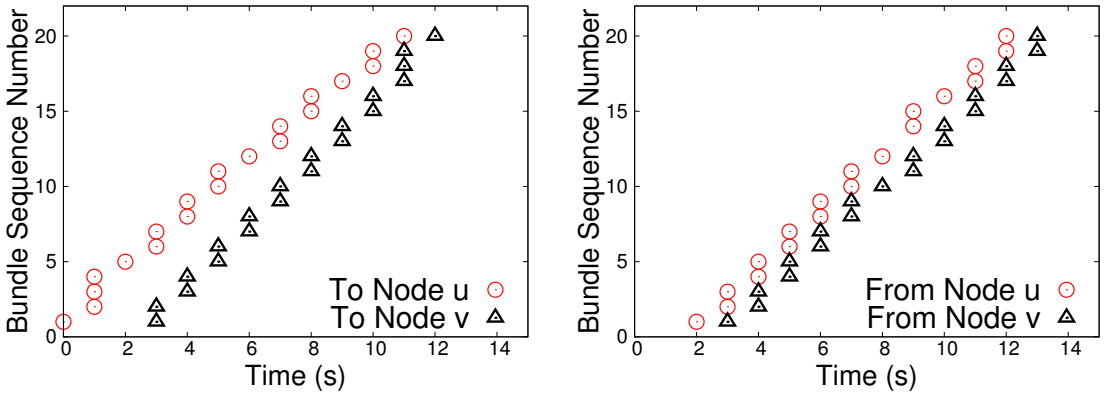


Figure 4.4: Synthetic trace results: (a) multi-link percentage; (b) duration of pair-link; (c) duration of 3-node-2-link; (d) duration of 3-node-3-link.

first turn on node v. After sufficient time, node u and v possess the same set of bundles. Next, we turned on node w. We collected node w's receiving times for each bundle from different sources (i.e., node u and v) and plot them in Figure 4.5b.

As we can see from Figure 4.5, when node w needed to transmit bundles to both u and v, it actually transmitted the same bundle twice. On the other hand, when both nodes u and v have the same bundles, both nodes try to transmit their copy of the same bundle to node w, since both have found that w did not possess that set of bundles. As a result, w received two copies of the same bundle and simply



(a) (b)

Figure 4.5: Timeline of node w. (a) sending timeline; (b) receiving timeline.

dropped the second one. These results generalize to other routing protocols that use replication, since they also do not consider multi-node contacts.

4.2 Transmission Scheduling for Multi-node Contact

In this section, we present our solution for addressing the problem of inefficient transmission in multi-node contacts.

4.2.1 Main Idea

Our main idea is to schedule transmissions and to use network coding during multi-node contacts, so that the number of transmissions can be reduced. One example is shown in Figure 4.6, where u has packet p_1 and w has packet p_2. Normally, it takes four transmissions to exchange p_1 and p_2. With network coding, node v transmits $p_1 \oplus p_2$ after receiving p_1 and p_2. u and w then decode the packet they need (p_2/p_1) using the packet they already have (p_1/p_2). Therefore, only three transmissions are sufficient. When replication is used during routing, it might be the case that v already has both p_1 and p_2, and only one transmission of $p_1 \oplus p_2$ would deliver both packets to u and w. Hence, the number of transmissions can be reduced by

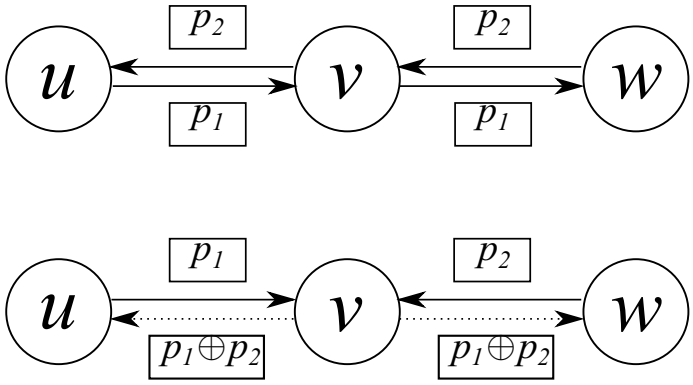


Figure 4.6: An example of network coding: u sends p_1 to w, while w sends p_2 to u.

designing an appropriate network coding scheme.

In this section, we only focus on optimizing transmissions in a multi-node contact; thus we assume that there already exists an opportunistic routing protocol that decides the next hop for each packet.

4.2.2 Network Model and Problem Formulation

We model a multi-node contact as a graph $G = (V, E)$, where V is the set of nodes and E is the set of pair-links. We assume that the topology of G does not change (no pair-link breaks) in T time slots and that the diameter of G is no larger than two hops. We focus on this model in this section and leave the problem for multi-node contact with larger diameters as future work. For each node v, we assume it is aware of all other nodes in G. Let $N(v)$ denote node v's one-hop neighbors, i.e., $N(v) = \{u : (u, v) \in E\}$. Each node v has a set of packets $P_v \subseteq P$, where $P = \cup_{v \in V} P_v$ is the set of all packets in this multi-node contact. For each packet $p \in P_v$, let $nexthop_v(p)$ denote the set of next hops of p decided by the routing protocol. Then for $p \in P$, we define $S(p) = \{v : p \in P_v\}$ as the set of nodes that already have the packet p, and $D(p) = \cup_{v \in V} nexthop_v(p)$ as the set of nodes that are the the next hops for packet p (as decided by the routing protocol).

We extend the "Want/Has" set model [12] to formulate our problem. We construct for each node v three set of packets, namely $W(v)$, $H(v)$, and $Aid(v)$. $W(v)$ is the "Want" set of node v, i.e., $W(v) = \{p : v \in D(p)\}$. $H(v) = P_v$ represents node v's "Has" set. We say that a packet $p \in W(v)$ is **satisfied** if v received p or p is decodable using the received coded packet and the packets in $H(v)$. $Aid(v)$ represents the "aid" set of node v, that is, node v "aids" to move $p \in Aid(v)$ to $u \in N(v)$ when $u \in D(p)$, u is not connected to any node in $S(p)$, and $p \notin W(v) \cup H(v)$. $Aid(v), v \in V$ is constructed in the following way: for a given packet p, if $u \in D(p)$ is not connected to any node v' that $p \in W(v') \cup H(v')$, then randomly select $v \in N(u) \cap \{\cup_{i \in S(p)} N(i)\}$ and insert p into $Aid(v)$. In each time slot, a node may transmit a packet or a coded packet. A coded packet is a combination of packets in transmitter's "Has" set, such that some packets in other node's "Want" set are satisfied.

Therefore, we formulate the problem as: minimize the number of transmissions to satisfy all packets in all "Want" sets $W(v), v \in V$. This problem is similar to index coding [12] but is still different in several ways. First, there is no central server that holds all the packets. Second, the network topology is not limited to a star-topology, where all the nodes are only connected to the server. Third, all nodes are able to send packets instead of only the central server sending the packets. These make the problem even harder than the index coding problem, proved to be an NP-hard problem. Therefore, we develop a heuristic method to solve it.

4.2.3 Contact Optimization Algorithm

In this section, we introduce our *Contact Optimization* algorithm to construct the transmission schedule. A schedule S consists of a series of tuples $\{(v, p), (v', p'), \dots\}$. Each tuple specifies the sender and the packet to be sent during a time slot.

The basic idea is to select a sender and a packet at each time slot t to maximize

the number of satisfied packets after this transmission. We select a tuple for each time slot until all packets in $W(v), v \in V$ are satisfied or a time limit T is reached. We present our algorithm in Algorithm 1.

We construct a series of graphs $G_v, v \in V$. Let $W'(v) = W(v) \cup Aid(v)$ represents the set of packets that node v does not have but are needed. First, for each node $w \in N(v)$ and each packet $p \in W'(w)$, we insert a vertex v_w^p to G_v, which has "Want" set $W(v_w^p) = \{p\}$ and "Has" set $H(v_w^p) = H(w)$ (Lines 5-8 in Algorithm 1). This way, all vertices in graph G_v will have the "Want" set with cardinality one. Next, we insert an edge between v_w^p and $v_{w'}^{p'}$ if: (1) $\{p, p'\} \subseteq H(v)$; and (2) one of the following conditions holds:

a. $p = p'$

b. $p \in H(v_{w'}^{p'})$ and $p' \in H(v_w^p)$

This is shown in Lines 9-13 of the algorithm. Note that if v_w^p and $v_{w'}^{p'}$ are connected in G_v, then one transmission from v (a coded packet $p \oplus p'$ or packet p if $p = p'$) can satisfy both packets in $W(v_w^p)$ and $W(v_{w'}^{p'})$, which actually satisfies p in $W'(w)$ and p' in $W'(w')$. If there exists a clique in G_v, i.e., a fully connected subgraph, then one transmission can satisfy all the packets in the "Want" set of nodes in that clique. Therefore, the maximum clique (denoted by max_clique) in all graphs G_v represents a packet that maximizes the number of satisfied packets for the given time slot. We denote the number of nodes in the maximum clique as $|max\_clique|$. We also use $p(max\_clique)$ to represent the packet constructed from max_clique, i.e., combination of packets in $\{p : v_w^p \in max\_clique\}$. Finding the maximum clique is presented in Lines 17-24 of the algorithm.

Algorithm 1 Contact Optimization Algorithm

Calculate each node v's $W(v)$, $H(v)$ and $Aid(v)$.
for all $v \in V$ **do**
    $W'(v) \leftarrow W(v) \cup Aid(v)$.
    Construct graph G_v:
    for all $w \in N(v)$, $p \in W'(w)$ **do**
        $W(v_w^p) \leftarrow \{p\}$, $H(v_w^p) \leftarrow H(w)$.
        Insert v_w^p to G_v.
    end for
    for all v_w^p, $v_{w'}^{p'} \in G_v$, and $w \neq w'$ **do**
        if ($p = p'$ or ($p \in H(v_{w'}^{p'})$ and $p' \in H(v_w^p)$) **then**
            Insert edge $(v_w^p, v_{w'}^{p'})$ to G_v.
        end if
    end for
end for
Initialize schedule $S = \emptyset$, time $t = 0$.
while (t $\leq$ T and not all packets are satisfied) **do**
    for Each G_v **do**
        $m \leftarrow MaxClique(G_v)$
        if ($|m| > |max\_clique|$) **then**
            $max\_clique \leftarrow m$
            $v_{sender} \leftarrow v$
        end if
    end for
    $S \leftarrow S \cup \{(v_{sender}, p(max\_clique))\}$
    for Each G_v **do**
        $G_v \leftarrow Update(G_v)$
    end for
    $t \leftarrow t + 1$
end while
Output S

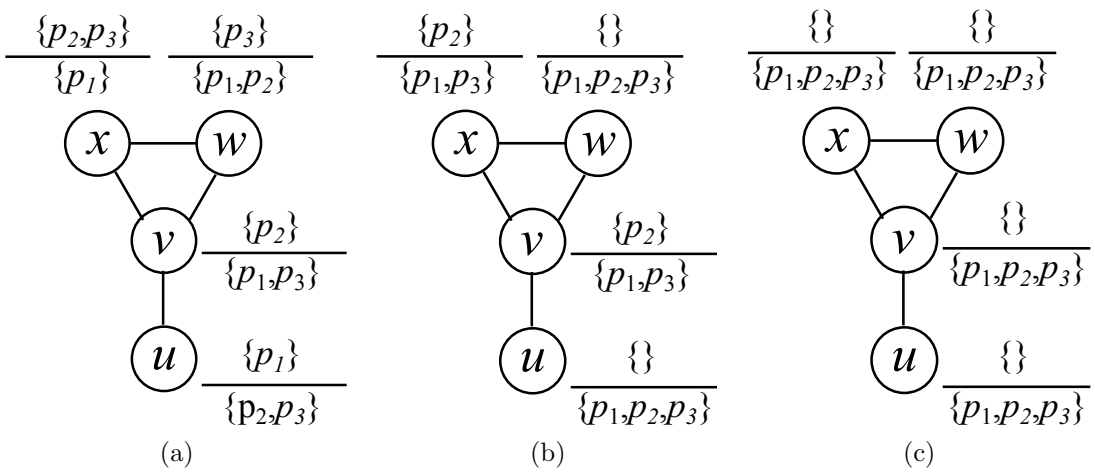


Figure 4.7: An example of transmissions with contact optimization algorithm.

Having decided the coded packet for slot t by finding max_clique, $G_v, v \in V$, the following need to be updated: (1) $v_w^p \in max\_clique$ should be eliminated from G_v since the packet in its "Want" set is satisfied; (2) Other nodes need to update their "Has" set: for all $v_w^{p'}$, $H(v_w^{p'}) \leftarrow H(v_w^{p'}) \cup \{p\}$ if $v_w^p \in max\_clique$; (3) New edges are formed between residual vertices if the two conditions are fulfilled. Consequently, they should be inserted in graph G_v.

4.2.4 Contact Optimization Example

To demonstrate the execution of our algorithm, consider a multi-node contact as shown in Figure 4.7. For simplicity, we assume the underlying routing protocol is Epidemic. For a given node, the "Has" set is constructed by inserting all the packets it already has. The "Want" set is constructed by inserting all the packets it does not possess and the "aid" set is empty due to Epidemic routing. We show each node's "Want/Has" set as $\frac{\{Want\}}{\{Has\}}$ in Figure 4.7. According to Algorithm 1, we select v to transmit a coded packet $p_1 \oplus p_3$ first, since it can satisfy three packets: p_3 (for x), p_3 (for w), p_1 (for u). Then we update each node's "Want/Has" set as shown in

Figure 4.7b. Next we select w to transmit packet p_2 since it can satisfy two packets: p_2 (for v) and p_2 (for x). Finally all the packets in "Want" sets are satisfied, as shown in Figure 4.7c, and the algorithm terminates.

4.3 Performance Evaluation

In this section we present the performance evaluation of our *Contact Optimization* algorithm. We use a trace-driven simulator. We chose **Epidemic** [82] and **Spray&Wait**[79] as routing protocols for all simulations. Spray&Wait is implemented with the binary spray scheme, with spray counter empirically set to half of the number of nodes. We implemented our *Contact Optimization* algorithm with both routing protocols, named as **Epidemic-ConOpt** and **SprayWait-ConOpt** respectively. We compare the results between **R** and **R-ConOpt**, where **R** refers to either Epidemic or Spray&Wait.

We performed simulations on real world mobility traces. For all simulations, we generated three data flows with sources and destinations chosen uniformly at random. All packets arrived at the source at time 0 and the packet size was set to 2.5MB. We assumed unlimited buffer for each node. The bandwidth was set to 20Mbps. For performance evaluation, we use the following three metrics:

- **Total number of transmissions**: This is the key metric that we are interested in, as it reflects whether our algorithm indeed reduces the number of transmissions and thus improve the transmission efficiency.

- **Average Packet Delivery Ratio (PDR)**: Although we do not optimize for the PDR, but the scheduling algorithm should not be detrimental to the overall PDR performance. In fact, as we implicitly increase the contact capacity, we expect there would be slightly improvement of the PDR of R-ConOpt.

- **Average Packet Delivery Delay (PDD)**: We evaluate PDD to test whether R-ConOpt is able to deliver packets earlier than the original routing although we do not explicitly optimize for it.

We evaluate our *Contact Optimization* algorithm using the Cambridge mobility trace. We chose not to use Intel and Infocom conference traces as Intel is too small while contacts in conference traces are too dense. The Cambridge bluetooth contact trace consists of 12 nodes with a duration of 5 days. We only consider contacts between the 12 users (as Bluetooth traces might indicate the nearby presence of other devices than the 12). We selected three days of the trace (we skipped the first day when devices were distributed to participants and weekends), with each day starting at 9 a.m. and ending at 18 p.m. (as normal working hours). Each day is further divided into three time windows with three hours for each time window, i.e., nine traces in total. For each mobility trace we generated 10 message traces. Each point in the performance evaluation result is the average over 90 runs initialized with random seeds. Since not all nodes are present in all the traces, we only choose existing nodes as sources/destinations for the data flows. We varied the network loads and evaluate the performance. The results are presented in Figure 4.8.

As we can see from Figure 4.8, R-ConOpt reduces the total number of transmissions by an average 13%. This result demonstrates the ability of *Contact Optimization* for reducing the number of transmissions. One can also observe that when the network load increases, the PDR decreases since the load already exceeds the network capacity. The PDD also increases when the network loads are increasing. It is important to notice that R-ConOpt is able to maintain a slightly better performance against the corresponding routing protocol without *Contact Optimization* algorithm, in terms of both PDR and PDD. This indicates that our algorithm is also able to

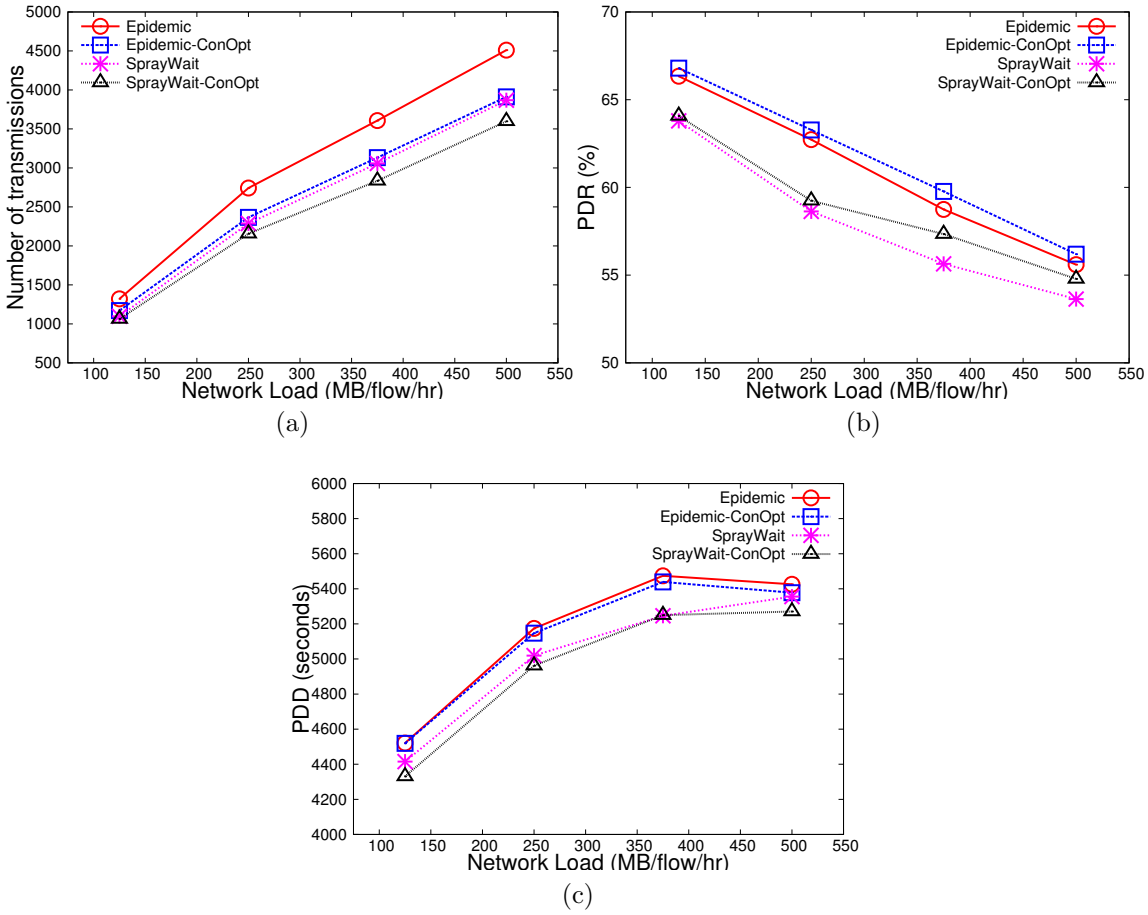


Figure 4.8: Cambridge mobility trace results: (a) total number of transmissions; (b) PDR; (c) PDD.

maintain the routing performance.

In order to further understand if and how node density influences the performance of our *Contact Optimization* algorithm, we present the results in terms of reduction of number of transmissions separately for three time windows: 9-12 p.m., 12-15 p.m., and 15-18 p.m. Here we only present results using Epidemic, as Spray&Wait has similar results. One important indicator of the node density is the *average node degree* of the network over time. We thus calculate the average node degree at each moment and plot it in Figure 4.9. As we can see from Figure 4.9, the average node

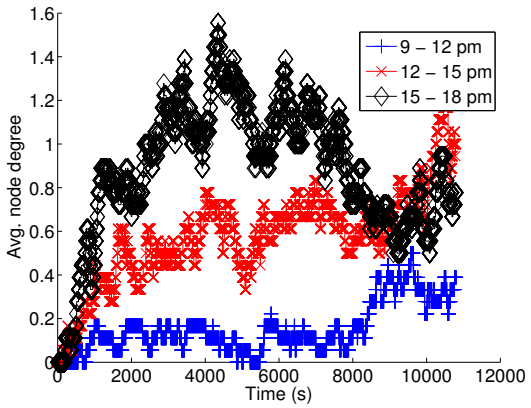


Figure 4.9: Average node degree for the Cambridge trace.

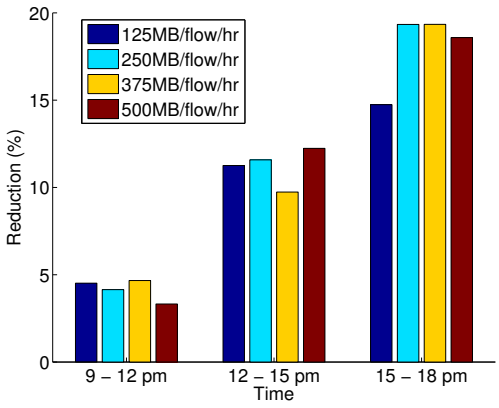


Figure 4.10: Reduction in total number of transmissions as a function of time of the day.

degree varies between these three time windows, with 15-18 p.m. being the largest. This indicates that nodes in these three days are better connected in the time window 15-18 p.m. It is important to notice that the reduction in packet transmissions also peaked during the 15-18 p.m. time window of 15-18 p.m. This reduction in transmissions is as high as 20%, as shown in Figure 4.10. This result illustrates that *Contact Optimization* is able to perform better in a network with good connectivity.

Intuitively this is true since better connectivity means more opportunities for network coding, and thus fewer transmissions.

5. ENERGY-BALANCED ROUTING PROTOCOL[1]

Opportunistic Mobile Networks (OMN) have gained increasing attention in recent years due to the widespread use of mobile cell phones that have powerful computing, sensing and communicating capabilities [15]. In OMNs, human held mobile devices communicate opportunistically due to the lack of end-to-end connectivity. Early research has focused on exploiting pairwise contact to disseminate information, while a recent study [92] has identified the wide existence of *Transient Connected Components* (TCCs) in OMNs and has revealed their positive impact on information dissemination.

Recent OMN routing protocols [17] [34] [48] achieve good delivery rate and delay performance with relatively low routing overhead by exploiting social structure of humans [76], such as identifying and utilizing "high quality" nodes. However, as a node with high quality metric is more likely to *receive a message and becomes a message carrier*, it inevitably carries a high *memory load* [23]. This results in higher buffer utilization, more received and transmitted messages, and consequently, a *higher energy consumption rate*. Using two real world mobility traces [22] [55], we show that the energy consumption imbalance problem indeed exists in social-based routing protocols. Moreover, protocols utilizing TCCs suffer from more severe memory load imbalance. Although utilizing popular nodes is critical for social-based routing protocols, over-utilization of these nodes may result in multiple detrimental effects on routing performance [76].

Existing OMN congestion control mechanisms [67] [46] [69] that are based solely

[1]Reprinted from "On balancing the energy consumption of routing protocols for opportunistic social networks" by Chen Yang, Radu Stoleru, 2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC), Nanjing, 2015, pp. 1-9. Copyright 2015 by IEEE.

on balancing buffer utilization do not guarantee energy consumption balance as popular nodes deliver messages much faster than typical nodes. This results in higher aggregated traffic flow and thus higher energy consumption. Moreover, even if popular nodes are not selected to carry messages, they still have a higher chance for relaying messages for other nodes within the same TCC when they are on shortest paths. As existing OMN congestion control mechanisms are TCC-unaware, they will suffer from this problem in networks with TCCs.

Due to the presence of TCC in OMNs, accurate analysis of energy consumption (stemming from node transmission/communication overhead) and finding good metrics for balancing energy consumption of routing protocols are extremely difficult. The main ideas we propose are the use of the node *memory load*, and the need for *energy-aware intra-TCC routing*. We propose a novel Memory Load-aware routing metric that combines both routing quality metric and memory load metric; and Energy-aware Intra-TCC routing to avoid hot zones within TCCs. Based on these, we propose our Energy Consumption Balanced Routing protocol that reduces energy consumption imbalance while maintaining routing performance. We analyze our protocol and show that after sufficient time, a node's average memory load rate converges, and implicitly the average energy consumption rate converges due to the demonstrated correlation between two of them. We evaluate our Energy Consumption Balanced Routing protocol using the Reality and UCSD mobility trace. Simulation results show that our protocol can significantly improve both energy balance and memory load balance while achieving comparable routing performance (PDR, PDD and data copy overhead) with existing social-based routing protocols. The contributions of this section are as follows: (a) it analyzes the energy consumption imbalance in OMNs with TCCs and demonstrates that TCC-aware routing protocols suffer from a more severe memory load imbalance; (b) it proposes an Energy Consumption

Balanced Routing protocol which provably guarantees the convergence of average memory load rate; (c) using real world mobility traces, it shows the effectiveness of proposed solutions when compared with state-of-art routing protocols.

5.1 Preliminaries

5.1.1 Transient Connected Component (TCC)

It is identified in [92] that, in addition to pairwise contacts, *Transient Connected Components* (TCCs) widely exist in OMNs. TCCs are temporally formed in OMNs where mobile devices are able to communicate in a multi-hop manner. Although the existence of TCCs increases the contact probability between nodes, we show that TCC-aware routing protocols may suffer from a more severe memory load imbalance problem, when compared to TCC-unaware routing protocols.

5.1.2 Social-based Routing Protocols

Most social-based routing protocols use different *quality metrics* to measure the contact capability of a given node. When a node encounters a new neighbor, these quality metrics are used by *forwarding strategies* to decide whether to forward a message or not. *Message carriers* are the nodes who hold the message for further forwarding (when TCC topology changes occur) or delivering to destinations. A node with high quality metric is therefore assumed to be a better *message carrier* than a node with lower quality metric.

5.1.2.1 Quality Metrics

The proposed quality metrics can be classified into two categories: *destination-independent* and *destination-dependent*. A destination-independent metric represents the contact capability of a node regardless of the destination of the message. This type of metric includes **Betweenness** [17] [34] and **CCP** [29]. On the other hand, a

node may have different quality metric values, depending on different destinations. For example, **Dest. Frequency** [23] and **Contact Probability** [48] are destination-dependent. We briefly introduce **Betweenness** and **Dest. Frequency** which are used in this section.

Betweenness: Consider a *contact graph* $G = (V, E)$ where an edge linking v_i and v_j means that v_i and v_j have met each other. The Betweenness c_i of node v_i is defined as: $c_i = \sum_j \sum_k \frac{g_{jk}(v_i)}{g_{jk}}$, where g_{jk} represents the total number of shortest paths from v_j to v_k, and $g_{jk}(v_i)$ is the number of shortest paths that contains node v_i.

Dest. Frequency: The destination frequency c_{ij} of node v_i to destination v_j is defined as $c_{ij} = \lambda_{ij}$, where λ_{ij} is the contact rate between node v_i and v_j.

5.1.2.2 Forwarding Strategies

We assume that the node who initiates the forwarding still keeps the message. We adopt the following forwarding strategies as they are widely used:

Compare and Forward: The node forwards the message to the encountered node if the latter one has a higher quality metric. When multiple nodes contact, the node selects the one with the highest quality metric. This strategy is widely adopted by previous research [34] [17] [48].

Delegation Forwarding: The node forwards the message to the encountered node if the latter one has the highest quality metric among all nodes it previously met. This strategy was proposed in [23] to further reduce the number of data copies.

5.1.2.3 Routing Protocols

The combination of a quality metric and a forwarding strategy results in a routing protocol. If the protocol only selects *message carriers* from its one hop neighbors, we say it is a TCC-unaware routing protocol. If the protocol selects *message carriers*

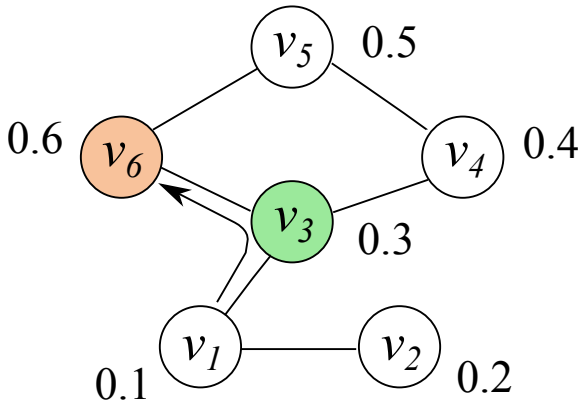


Figure 5.1: An example of routing decision difference between TCC-unaware and TCC-aware routing protocols.

from all nodes within a TCC, we say it is a TCC-aware routing protocol. An example is shown in Figure 5.1, where the quality metric is shown beside each node. In this example, if v_1 uses a TCC-unaware protocol, it chooses the *next carrier* from the set $\{v_2, v_3\}$ and hence v_3 will be chosen. If it uses a TCC-aware protocol, it chooses from the set $\{v_2, \ldots, v_6\}$ and hence v_6 will be chosen. Notice that when v_6 is selected as the *next carrier*, intermediate node v_3 only *relays* the message to v_6 without adding it to its own buffer, i.e., v_3 will not become a message carrier. When the selected *next carrier* is multiple hops away, we assume that the node which initiates the forwarding calculates the shortest path and performs source routing. In the rest of the section, we reserve the term "forwarding" for OMN forwarding, while using "relaying" for intra-TCC forwarding.

5.1.3 Datasets

In this section we use two mobility traces for both analysis and performance evaluation. The details are shown in Table 5.1. As shown, the *MIT Reality* [22] trace consists of 97 users carrying cell phones. Users are selected from MIT students, staff and faculty members. The mobility trace contains Bluetooth sightings

Trace	*MIT Reality*	*UCSD*
Device	Phone	PDA
Network type	Bluetooth	WiFi
No. devices	97	275
Duration (days)	246	77

Table 5.1: Mobility traces used for the analysis and evaluation of the Energy-balanced Routing protocol

between devices over the course of nine months. The *UCSD* [55] have recorded the Access Point detection and association events for 275 hand-held PDAs distributed to students. The trace covers eleven weeks. We assume that there is a contact event between two users when the time windows in which they detect the same AP overlap.

5.2 Motivation and Memory Load Analysis

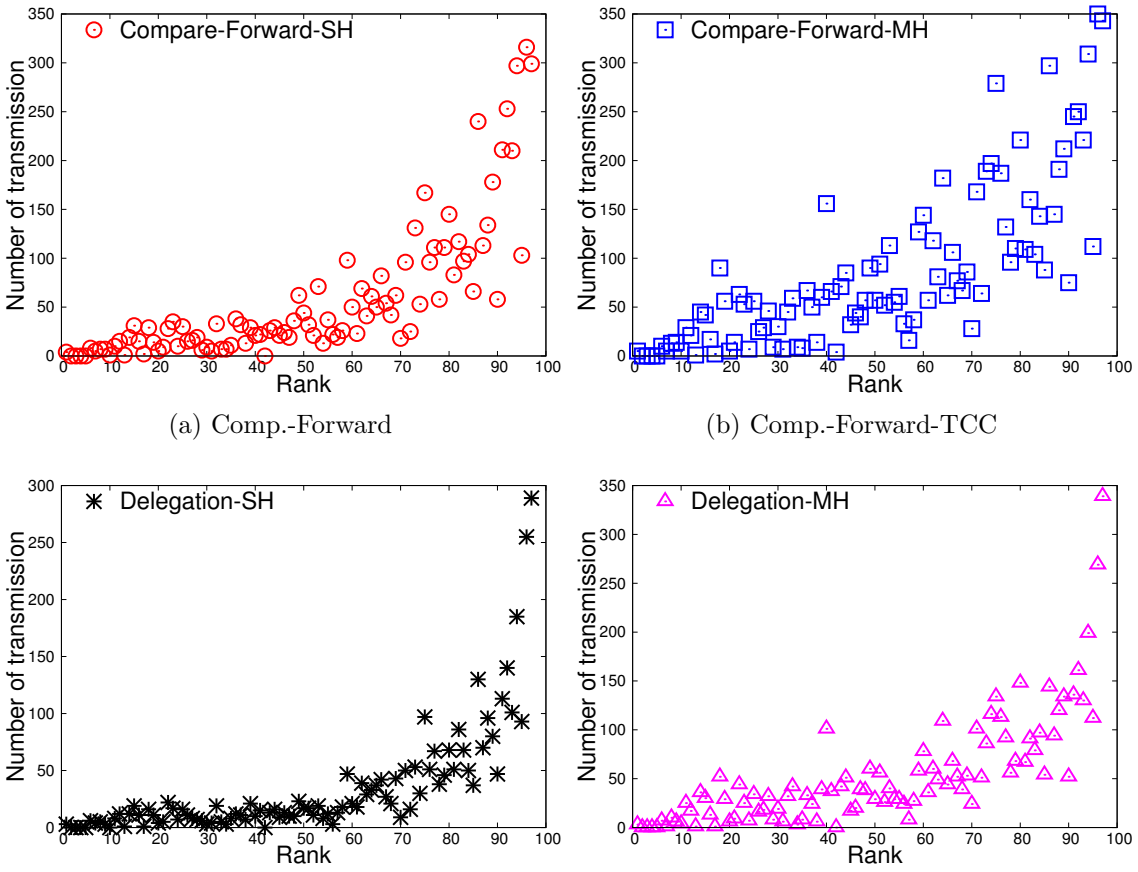
To motivate our research, we perform simulations for existing social-based routing protocols on two real world mobility traces. These empirical results demonstrate that a severe energy consumption imbalance indeed exists. As mentioned in Section 1, we argue that memory load [23] is a key indicator of energy consumption. We therefore theoretically analyze memory load, as an accurate analysis of energy consumption is difficult due to the presence of TCC (note that in a TCC, different routes exist between any pair of nodes). Our analysis shows that memory load is also severely imbalanced, and that routing protocols that utilize TCC suffer from more severe imbalance. We then validate our analytical results using simulations on real world traces.

5.2.1 Motivation

Transmission and reception of messages is the main source for energy consumption in OMN/DTN routing protocols [79]. The transmission and reception opportunity in an OMN with TCC mainly occurs in the following three scenarios: (a) a node

in a TCC has a message and decides to forward that message to another node in the TCC (i.e., the next carrier or the destination); (b) a node in the TCC relays a message (initiated by a node, as explained in (a)) on a path, towards the next carrier (as decided by the initiating node) or the destination; (c) a node is selected as a message carrier or is the destination and thus receives a message. In case (a) nodes with higher quality metric have a higher chance for forwarding messages to destinations (since they receive more messages) but a lower chance for forwarding messages to the next carrier (since the number of nodes with even higher metric is small). In case (b) a node with higher quality metric has a lower chance to serve as an intermediate node, since it has a lower chance to meet other nodes that have higher quality metric than itself. On the other hand, a node with a lower quality metric has a higher chance to serve as an intermediate node when it is in a TCC. But since the node has a low quality metric, it does not meet other nodes as often as nodes with high quality metric (if the metric reflects the contact rate). Furthermore, whether a node is on a forwarding path in the TCC highly depends on the network topology, which is even harder to analyze or predict. In case (c) nodes with higher quality metric have higher chance to be selected as message carriers. Given the complexity of analyzing the transmission imbalance, we perform simulations to understand the energy consumption imbalance.

We perform simulations using two mobility traces, Reality and UCSD. **Betweenness** is used as the quality metric. We pre-process the trace: we calculate each node's Betweenness metric and use it to rank nodes. 1,000 messages are generated with source and destination randomly chosen among all nodes within each trace. The deadline of each message is set to 24 hours. Once the message is delivered to the destination, it is eliminated from all nodes' buffers. We assume infinite communication bandwidth and buffer size. For each node, we record the total number of



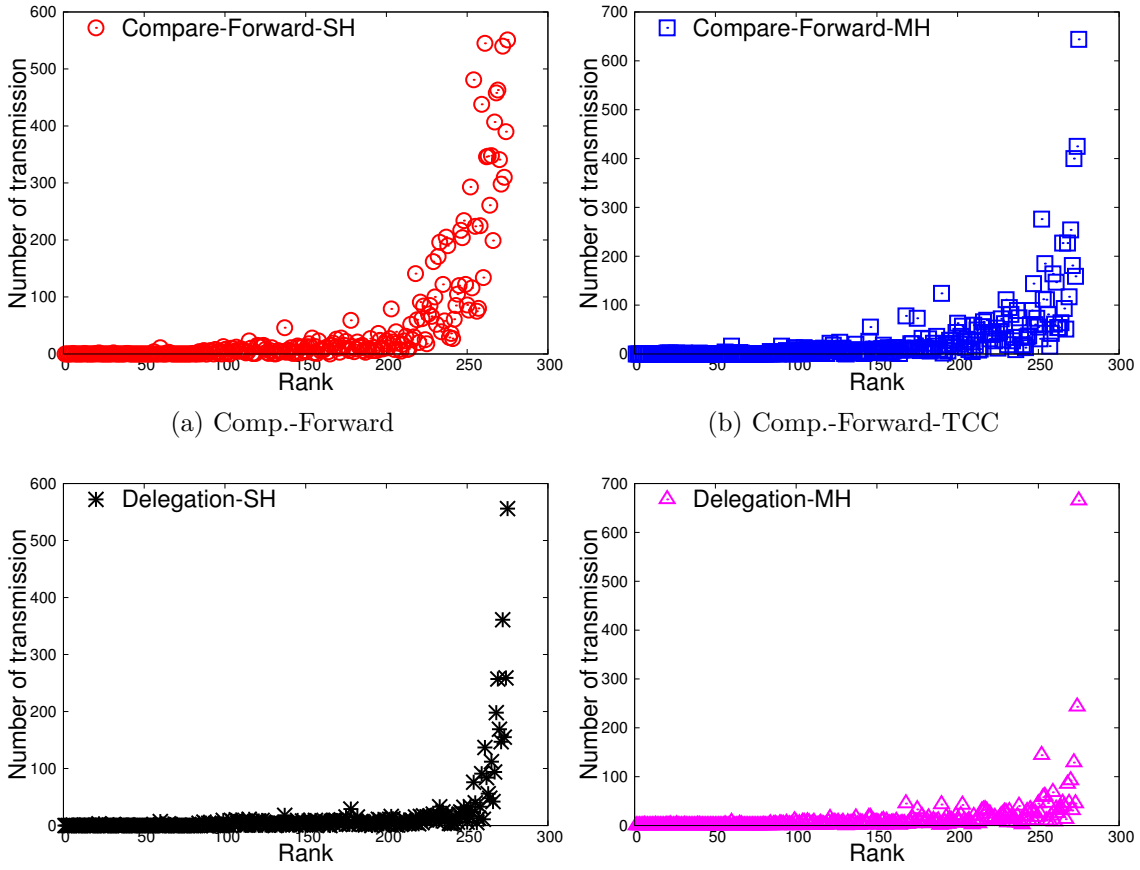
(a) Comp.-Forward (b) Comp.-Forward-TCC

(c) Delegation (d) Delegation-TCC

Figure 5.2: Energy consumption imbalance for two forwarding strategies in the MIT Reality trace: TCC-unaware (a,c), TCC-aware (b,d).

packet transmissions and receptions. We plot the *(rank, value)* pair for each node in Figure 5.2 and Figure 5.3. We use "-TCC" to indicate TCC-aware routing.

From Figure 5.2 and Figure 5.3, we can observe that the energy consumption imbalance exists. For TCC-unaware protocols, as shown in Figures 5.2a, 5.2c and Figures 5.3a, 5.3c, nodes with the top 10% Betweenness contribute up to 42% (74%) of the total energy consumption in the Reality (UCSD) trace. Although nodes within TCCs may serve as intermediate nodes and relay messages, the energy consumption imbalance for TCC-aware routing is still severe in spite of the added randomness. As



(a) Comp.-Forward (b) Comp.-Forward-TCC

(c) Delegation (d) Delegation-TCC

Figure 5.3: Energy consumption imbalance for two forwarding strategies in the UCSD trace: TCC-unaware (a,c), TCC-aware (b,d).

shown in Figures 5.2b, 5.3b and Figures 5.2d, 5.3d, the top 10% nodes contribute up to 32% (57%) of the total energy consumption in the Reality (UCSD) trace. These results motivate our research for reducing the energy consumption imbalance.

We propose that a key indicator of energy consumption is memory load. Next, we present our theoretical analysis and empirical results for memory load imbalance, and demonstrate its correlation with energy consumption.

5.2.2 Memory Load Imbalance Analysis

5.2.2.1 Network Model and Assumptions

Consider a network with a set of N nodes. Each node v_i has a quality metric c_i. We model the contact process of the entire network as a homogeneous Poisson process $\{T_n\}_{n\geq 0}$ [65], where $\{T_n - T_{n-1}\}_{n\geq 1}$ are i.i.d exponential variables with mean $1/\lambda$. During each contact event at time T_n, a set of node V_n meet with each other, where $|V_n| = S_n$. Each node v_i generates messages according to a renewal process $\{M_n^i\}_{n\geq 0}$, independent of the contact process, with mean inter-arrival time $1/\mu_i$, and message size $\{D_n^i\}_{n\geq 0}$. Each message has a deadline L, i.e., the n-th message is expired after $M_n^i + L$. We consider Compare and Forward as the forwarding strategy since it is the simplest and commonly used strategy. During each contact, only the node with the highest metric may receive a copy of the message.

We make the following assumptions to facilitate our analysis. We assume that c_i are distributed in $(0, 1]$ interval [23]. Notice that it is usually the relative order of metrics between two nodes that decides the message forwarding. So this assumption does not affect the analysis of load imbalance. Without loss of generality, let c_i be the i-th least metric, i.e., $c_1 < \cdots < c_N$. The duration of each contact is assumed to be negligible but sufficient for all data transfer, which is a common assumption in many DTN analysis [65]. S_n is shown to follow a geometric distribution [92]. We can therefore vary $E[S_n]$ to analyze TCC-aware and TCC-unaware routing protocols. We assume that the data sizes $\{D_n^i\}_{n\geq 0}$ are i.i.d random variables and are independent of both $\{M_n^i\}_{n\geq 0}$ and $\{T_n\}_{n\geq 0}$.

5.2.2.2 Memory Load Analysis

We model the memory load of a node v_j incurred by messages from v_i as a renewal reward process. For the n-th message from v_i, the "reward" $R_{i,j}^n$ for v_j

is a random variable $\mathbb{1}_{\delta_{i,j}^n} D_n^i$, where $\mathbb{1}_{\delta_{i,j}^n}$ indicates whether v_j receives a copy of the message before it expires. The total reward, i.e., the memory load, from v_i at time t is therefore $R_{i,j}(t) = \sum_{n=0}^{N_i(t)-1} R_{i,j}^n$, where $N_i(t) = \max\{n : M_n^i < t\}$ is the number of generated messages before t. The total memory load of node v_j is $R_j(t) = \sum_{i:i<j} R_{i,j}(t)$. By Elementary Renewal Theorem [72], we know that $\lim_{t\to\infty} E[R_{i,j}(t)]/t = \mu_i E[R_{i,j}^n] = \mu_i P(\delta_{i,j}^n) E[D_n^i]$. Therefore, the time average of the memory load of v_j is

$$\bar{R_j} = \lim_{t\to\infty} \frac{1}{t} E[R_j(t)] = \sum_{i:i<j} \mu_i P(\delta_{i,j}^n) E[D_n^i] \tag{5.1}$$

Now we compute the probability of event $\delta_{i,j}^n$. Notice that since $\{T_n\}_{n\geq 0}$ has intensity λ, the number of contacts N_C during a message's lifetime L is a Poisson random variable with mean λL. Therefore, $P(\delta_{i,j}^n | N_C) = 1 - \prod_{k=1}^{N_C}(1 - P(\delta_{i,j,k}^n))$, where $\delta_{i,j,k}^n$ denotes the event of *v_j receives a copy of v_i's n-th message during the k-th contact (with set $V_{(k)}$) after the message generation.* Notice that $\delta_{i,j,k}^n$ occurs, if both v_i and v_j are in $V_{(k)}$ and v_j has the highest metric. Assuming that each node v_i has probability p_i to be in the set V_k, then if N is sufficiently large, $P(\delta_{i,j,k}^n | S_{(k)}) = p_i p_j (1 - p_i - \sum_{m\geq j}^{N} p_m)^{S_{(k)}-2}$, for $S_{(k)} \geq 2$. We consider the case where the probability p_i is proportional to the node's quality metric, i.e., $p_i = c_i / \sum_l c_l$. This corresponds to many quality metrics where the node with a high total contact rate has a high quality metric [23]. $P(\delta_{i,j,k}^n)$ and $P(\delta_{i,j}^n)$ can then be calculated using total probability formula given the distribution of N_C and $S_{(k)}$, i.e., $P(\delta_{i,j,k}^n) = \sum_{l=2}^{\infty} Pr(S_{(k)} = l) p_i p_j (1 - p_i - \sum_{m\geq j}^{N} p_m)^{l-2}$, $P(\delta_{i,j}^n) = \sum_{l=0}^{\infty} Pr(N_C = l)(1 - \prod_{k=1}^{l}(1 - P(\delta_{i,j,k}^n)))$. Equation 5.1 can be evaluated when μ_i and $E[D_n^i]$ are known.

As a case study, let us assume that all nodes generate messages at the same rate and with the same expected data size, i.e., $\mu_i = 1$, $E[D_n^i] = 1$ for all i. Node v_i has a

	CnF-SH	CnF-MH	Del-SH	Del-MH
Reality	0.95	0.79	0.96	0.81
UCSD	0.94	0.90	0.96	0.94

Table 5.2: Correlation coefficient r between energy consumption and memory load

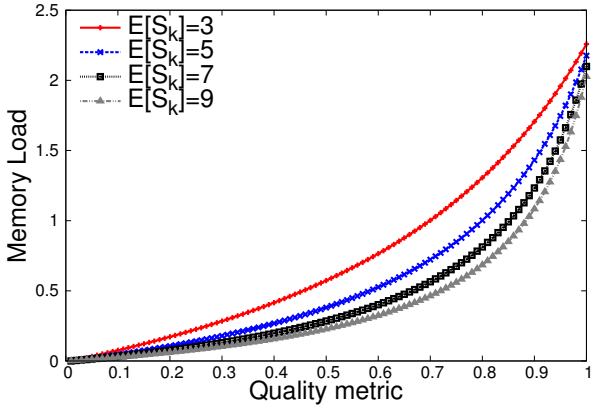


Figure 5.4: Theoretical analysis results for memory load imbalance

quality metric $c_i = i/N$, for $i = 1, \ldots, N$. Suppose λ is 10 times μ_i, i.e., $\lambda = 10$; and that L is 24 times inter-message event time, i.e., $L = 24$. We plot the time average of memory load $\bar{R}_j$ for each node v_j in Figure 5.4. It is clear that the memory load imbalance exists. Nodes with high quality metric incur much higher memory load than nodes with low quality metric. Moreover, and more importantly, we notice that the imbalance is increased with the increase of $E[S_k]$, i.e., the mean size of the encounter node set. *Since in our analysis a larger encounter node set mimics TCC-aware routing behavior, we argue that TCC-aware routing protocol results in more severe memory load imbalance.*

5.2.2.3 Empirical Results

We run simulations using parameters introduced previously. In addition to Be-

tweenness, we also use **Dest. Freq.** as quality metric. We record the value of memory load and plot $(rank, value)$ pairs. For Dest. Frequency, we use average rank to represent each node.

Results for Betweenness are shown in Figure 5.5a and Figure 5.5b. We can see that the memory load increases with the node's Betweenness metric, and that the trend fits our analytical results. The node with high Betweenness metric receives many more messages, when compared to typical nodes. Furthermore, when comparing the same forwarding strategy with different TCC-awareness, we can see that *TCC-aware routing makes the memory load imbalance more severe.* This again fits our analysis in the previous section.

Table 5.2 presents the Pearson correlation coefficient r between energy consumption and memory load. It is clear that they are positively correlated. We therefore consider memory load as an indicator for energy consumption.

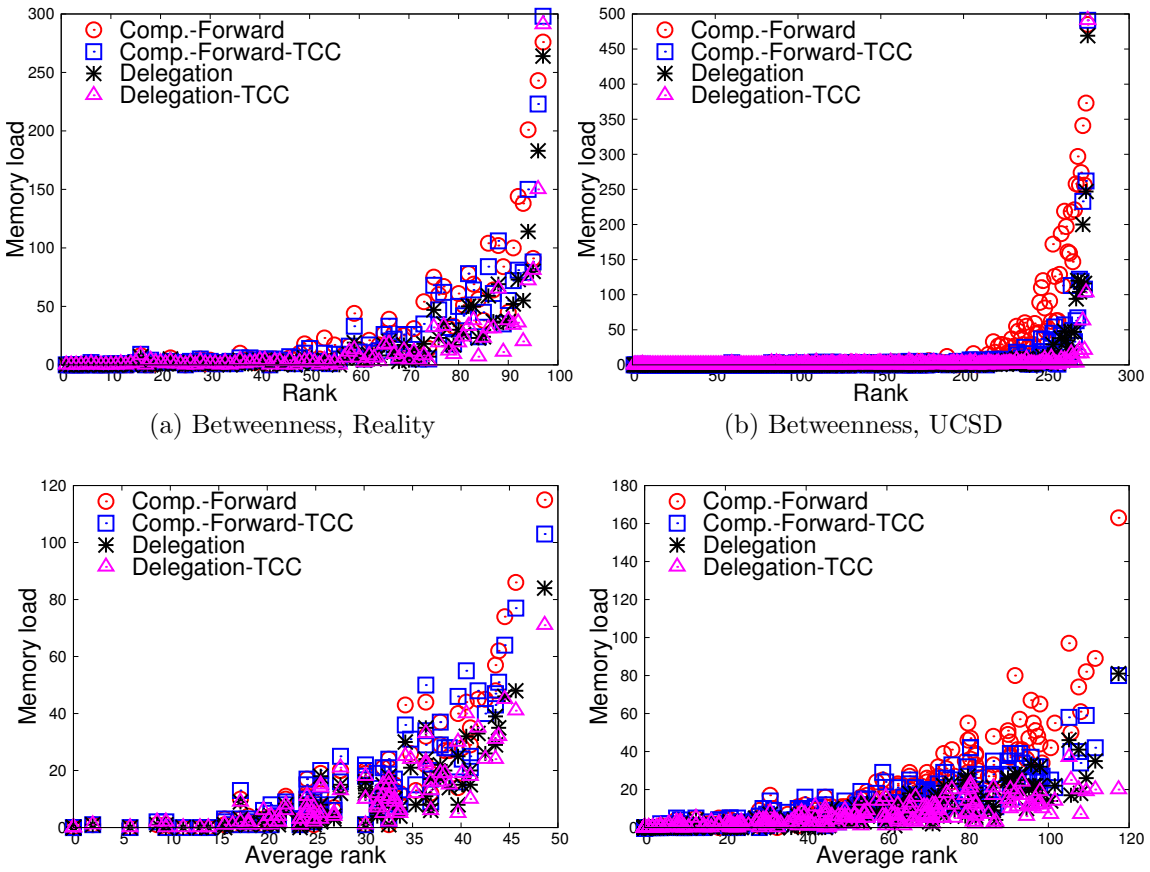
In Figure 5.5c and Figure 5.5d we can see that when using destination-dependent metric, nodes with high average contact rates receive many more messages. Since destination-dependent metrics have similar imbalance performance, we focus on destination-independent metrics in the rest of this section.

5.3 Energy Consumption Balanced Routing in Opportunistic Mobile Networks

In this section, we present our Energy Consumption Balanced Routing protocol for OMN with TCCs. We consider routing a single message m from source v_s to destination v_d. The message has a generation time t_m and a deadline T, i.e., the message expires after time $t_m + T$.

5.3.1 Main Idea

Our main ideas are to give higher priority to under-utilized nodes when selecting message carriers and to perform intra-TCC routing in an energy balanced manner.



(a) Betweenness, Reality

(b) Betweenness, UCSD

(c) Dest. Freq, Reality

(d) Dest. Freq, UCSD

Figure 5.5: Memory load imbalance for two quality metrics in the MIT Reality trace (a,c) and the UCSD trace (b,d)

We jointly consider both *quality* metric and *memory load quota* metric when selecting message carriers, as we demonstrated in the previous section that memory load is a key indicator for energy consumption in OMN. The quality metric reflects the capability of a node to deliver the message within a short period of time. Always selecting high quality metric nodes as message carriers can maximize the PDR but results in severe energy consumption imbalance. We propose a *memory load quota* metric to indicate if the node is over-utilized or under-utilized. If a node is already over-utilized, we avoid selecting it as message carrier in the near future. We propose

a Memory Load-aware routing metric by combining both metrics, which is then used as input to the TCC-aware Comp. and Forward strategy. We also propose to use *Energy-aware Intra-TCC routing* to route messages within a TCC. As popular nodes have a greater chance to lie on shortest paths between other nodes in a TCC, it is necessary to avoid selecting paths that contain over-utilized nodes when routing within a TCC. We present the details of our protocol in the following sections.

5.3.2 Memory Load-aware Routing Metric

We associate with each node v_i a *memory load quota* q_i. At time $t = 0$, q_i are set to q_{init} for all v_i. When v_i receives a message and adds the message to its buffer, q_i is updated by $q_i \leftarrow q_i - q_{cost}$ where q_{cost} is a constant. We denote the *memory load quota* of v_i at time t as $q_i(t)$. Notice that $q_i(t)$ reflects the utilization of node v_i until moment t. If node v_i has a significantly lower $q_i(t)$ when compared to other nodes, we consider that it is over-utilized.

Now we define the Memory Load-aware routing metric.

Definition 5. *The Memory Load-aware routing metric $r_i(t)$ for node v_i at time t is: $r_i(t) = c_i + \rho \cdot q_i(t)$, where $\boldsymbol{c_i}$ is node $\boldsymbol{v_i}$'s quality metric, $\rho > 0$ is a control variable.*

When routing a specific message m, we use a *message-specific Memory Load-aware routing* metric, defined by

$$r_{i,m} = r_i(t_m) = c_i + \rho \cdot q_i(t_m)$$

where t_m is the generation time of message m.

Notice that $r_{i,m}$ is an evaluation of $r_i(t)$ at moment t_m. We choose to use a message-specific metric to reduce the number of data copies. In fact, if we evaluate the metric at the exact moment when making forwarding decisions, unnecessary

data copies may be created to balance the memory load among nodes. We show through simulations that our protocol has comparable data copy overhead with other protocols.

In order to keep track of q_i, each node has to maintain a *memory load quota table*, which includes a series of $(t, q_i(t))$ pairs. Entries are purged if they are older than a certain threshold, which we set to the message's deadline T.

5.3.3 BetRank Metric

In this section we propose a new node quality metric called *BetRank*.

Definition 6. *BetRank c_i for node v_i: Suppose node v_i has the k-th least Betweenness among N nodes, then its BetRank is defined as $c_i = k/N$.*

Notice that on one hand BetRank preserves the relative order of Betweenness metric between different nodes. Thus BetRank and Betweenness will have identical routing performance. On the other hand, BetRank enforces node's quality metric to be uniformly distributed in $(0, 1]$. We will show, through analysis, that when Memory Load-aware routing metric uses BetRank as node quality metric, the memory load consumption rate for different nodes converges.

In OMNs, the Betweenness metric can be estimated through the Ego network betweenness [17]. Here, we introduce a simple algorithm to accurately estimate the rank of each node's metric, as shown in Algorithm 2. Each node maintains a list of timestamped Betweenness values of all nodes, including itself, and periodically disseminates this information to its neighbors. By updating this information (line 4 to 10), each node counts the number of nodes that have smaller Betweenness metric than itself (line 11 to 15), which leads to an estimation of rank. A tie can be broken by the node ID. Due to space limitation, the evaluation of BetRank is in [88].

Algorithm 2 BetRank Calculation For v_i.

```
1: Upon receiving timestamped node metric list L
2: Update myBet
3: k ← 0
4: for all entry in L do
5:     myEntry ← myList.get(entry.nodeID)
6:     if myEntry.t < entry.t then
7:         myEntry.bet = entry.bet
8:         myEntry.t = entry.t
9:     end if
10: end for
11: for all entry in myList do
12:     if entry.bet < myBet then
13:         k ← k + 1
14:     end if
15: end for
16: myBetRank = k/N
```

5.3.4 Energy-aware Intra-TCC Routing

Although controlling memory load is key to energy consumption balance, it is not sufficient for a TCC-aware routing protocol. In a TCC-aware routing protocol, a node that initiates forwarding has to find a path to the next message carrier. Controlling memory load keeps popular nodes from constantly being selected as message carriers. However, popular nodes that have good connectivity may always lie on the shortest path between other nodes within TCC. Energy imbalance may also incur, as popular nodes have to relay more messages.

To address this problem, we propose an *Energy-aware Intra-TCC routing*. We represent a TCC as a graph $G = (V, E)$, where V is the set of nodes, and E is the set of edges. Suppose that each node v_i has consumed energy $e_c(i)$. We associate with each edge $(v_i, v_j) \in E$ a weight $w_{ij} = \frac{1}{2}(e_c(i) + e_c(j))$. Then the shortest path algorithm applied to the weighted graph prefers nodes with high remaining energy,

Algorithm 3 Energy Consumption Balanced Forwarding Algorithm For v_i.

```
1: for all message m in v_i's message queue do
2:     if m.Dest ∈ TCC then
3:         m.Path ← EnergyAwareIntraTCC(v_i, m.Dest)
4:         v_i.Send(m)
5:     else
6:         Find r_{j,m} = max{r_{k,m}, v_k ∈ TCC}
7:         if r_{j,m} > r_{i,m} and m ∉ v_j's msg. queue then
8:             m.Path ← EnergyAwareIntraTCC(v_i, v_j)
9:             v_i.Send(m)
10:        end if
11:    end if
12: end for
```

and thus avoids nodes that have already been over-utilized.

5.3.5 Energy Consumption Balanced Routing Protocol

Here we present our Energy Consumption Balanced Routing protocol, which mainly consists of the following three parts.

5.3.5.1 Beacon Exchange

In addition to neighbor discovery, each node broadcasts a *beacon* periodically. Each *beacon* includes a timestamped metric list, consumed energy, neighbor list, message list and a *memory load quota table*. By collecting other node's beacons, each node is able to maintain the topology of the current TCC and learn other's local information.

5.3.5.2 Message Forwarding

When a node v_i detects a TCC topology change, it invokes the forwarding algorithm presented in Algorithm 3. The node first checks if a given message can be delivered to the destination (line 2 to 4). If not, it then finds the node with the maximum Memory Load-aware routing metric in the current TCC (line 6). If the

found node has higher metric than itself and does not have the message, it forwards the message to the found node (line 7 to 10). If the selected message carrier or the destination is multiple hops away in the same TCC, the node performs *Energy-aware Intra-TCC routing* as presented in Section 5.3.4.

5.3.5.3 Message Receiving

When a node v_i receives a message, then: (a) if it is the final destination of the message, it simply adds the message to its received message queue; (b) if it is the intermediate destination (i.e., a selected message carrier) of the message, the node adds the message to its local buffer, updates q_i, and adds a new entry $(t, q_i(t))$ to its *memory load quota table*; (c) if it is the intermediate hop, the node extracts the next hop from the message and transmits the message.

5.3.6 Routing Protocol Analysis

In this section we analyze our Energy Consumption Balanced Routing protocol. We show that our protocol leads to the convergence of the memory load quota consumption rate for nodes with different quality metrics.

Since we assume that all nodes have initial memory load quota $q_i(0) = q_{init}$, then $r_i(0) = c_i + \rho \cdot q_{init}$. $r_i(0)$ and c_i have the same rank for v_i. However, since the memory load quota consumption rates (which we call quota consumption rate) are different, $r_i(t)$ has different ranks at time $t > 0$. We denote the rank of $r_i(t)$ among N nodes as $\phi_i(t)$.

We notice that given the same message generation rate, memory load is mainly decided by the rank of the metrics. Therefore, if a node's metric c has rank x, its memory load is a function of x, which we denote as $f_r(x)$. The quota consumption rate $g(x)$ is then proportional to the memory load, i.e., $g(x) = \alpha \cdot f_r(x)$, where $\alpha > 0$ is a constant. Since $f_r(x)$ is an increasing function, $g(x)$ is also an increasing

function. Then, the quota consumption rate $\sigma_i(t)$ for node v_i at time t is $\sigma_i(t) = g(\phi_i(t)) = \alpha \cdot f_r(\phi_i(t))$ and the average quota consumption rate $\overline{\sigma}(t)$ for all nodes at time t is:

$$\overline{\sigma}(t) = \frac{1}{N}\sum_{i=1}^{N}\sigma_i(t) = \frac{1}{N}\sum_{i=1}^{N}g(\phi_i(t)) = \frac{1}{N}\sum_{i=1}^{N}g(i) = \overline{\sigma} \tag{5.2}$$

where the third equality is due to $\{\phi_i(t), i = 1, \ldots, N\} = \{1, 2, \ldots, N\}$, and $\overline{\sigma}$ is a constant.

Now we consider the average quota consumption rate for a given node v_i, defined as $\overline{\sigma_i}(t) = \frac{1}{t}\int_0^t \sigma_i(\tau)\mathrm{d}\tau$. We will show that $\overline{\sigma_i}(t)$ converges to average quota consumption rate (i.e., $\overline{\sigma}$) for all nodes.

The Memory Load-aware routing metric for node v_i is $r_i(t) = c_i + \rho \cdot q_{init} - \rho \cdot \int_0^t \sigma_i(\tau)\mathrm{d}\tau$. Notice that the change of rank among different $r_i(t)$ depends on the third term in the above equation, as the second term is the same for all nodes.

Since we assume that c_i are uniformly distributed in $(0, 1]$, after time $t = 0$ the first change of rank only occurs between the nodes v_{N-1} and v_N as $r_N(t)$ decreases the fastest (with rate $-\rho \cdot g(N)$). Therefore, we first consider node v_{N-1} and v_N at time $t = 0$. At time $t = 0$, node v_{N-1} and v_N's metric $r_{N-1}(0)$ and $r_N(0)$ have the rank $N - 1$ and N, respectively. Consider a small time interval $[0, \delta t)$, during which the rank is not changed. Then we have:

$$r_{N-1}(\delta t) = c_{N-1} + \rho \cdot q_{init} - \rho \cdot g(N-1) \cdot \delta t$$
$$r_N(\delta t) = c_N + \rho \cdot q_{init} - \rho \cdot g(N) \cdot \delta t$$

Since $r_N(\delta t)$ is decreasing at a higher rate, at time δt_1, we have $r_{N-1}(\delta t_1) = r_N(\delta t_1)$, where δt_1 is given by $\delta t_1 = \frac{c_N - c_{N-1}}{\rho \cdot (g(N) - g(N-1))}$. After δt_1, node v_{N-1} and v_N can only have the same quota consumption rate: $\frac{1}{2} \cdot (g(N-1) + g(N))$.

Then since $r_{N-1}(t)$ and $r_N(t)$ are decreasing together at a rate higher than $r_{N-2}(t)$, we have $r_{N-2}(t) = r_{N-1}(t) = r_N(t)$ at time δt_2, which is given by:

$$\delta t_2 = \frac{\frac{1}{2}(c_N + c_{N-1}) - c_{N-2}}{\rho \cdot (\frac{1}{2}(g(N) + g(N-1)) - g(N-2))}$$

If we denote $\tilde{c}_k = \frac{1}{N-k+1}\sum_{i=k}^{N} c_i$ and $\tilde{g}(k) = \frac{1}{N-k+1}\sum_{i=k}^{N} g(i)$ for $k = 1, \ldots, N-1$, while $\tilde{c}_N = c_N$ and $\tilde{g}(N) = g(N)$. By observing the pattern, we can write the expression for δt_{N-i} for node v_i as:

$$\delta t_{N-i} = \frac{\tilde{c}_{i+1} - c_i}{\rho \cdot (\tilde{g}(i+1) - g(i))},\ i = 1, \ldots, N-1$$

Therefore, node v_i has quota consumption rate $\sigma_i(t) = g(i)$ in time interval $[0, \delta t_{N-i}]$, and $\sigma_i(t) = \tilde{g}(k)$ in time interval $(\delta t_{N-k}, \delta t_{N-k+1}]$ for $k = i, i-1, \ldots, 2$. After time δt_{N-1}, every node has the same quota consumption rate $\overline{\sigma}$.

Let $\delta t_0 = \delta t_1$, we can write the average quota consumption rate $\overline{\sigma_i}(t)$ for node v_i for sufficiently large t as

$$\frac{1}{t}\left(g(i)\delta t_{N-i} + \sum_{k=2}^{i} \tilde{g}(k)(\delta t_{N-k+1} - \delta t_{N-k}) + \int_{\delta t_{N-1}}^{t} \overline{\sigma}\mathrm{d}\tau\right)$$

for $i = 1, \ldots, N$. As the first two terms are finite, we have

$$\lim_{t\to\infty} \overline{\sigma_i}(t) = \overline{\sigma}$$

This result demonstrates that after sufficient time, all nodes have the same quota consumption rate. *In the next section we will show that the energy consumption imbalance will be reduced, due to the convergence of quota consumption rate and its strong correlation with energy consumption.*

5.4 Performance Evaluation

We evaluate our Energy Consumption Balanced Routing protocol through simulations using two mobility traces (MIT Reality and UCSD), and a custom trace-driven simulator. For UCSD, we use the first two weeks to reduce simulation time. We assume infinite communication bandwidth and infinite buffer size for each node. 50,000 and 10,000 messages are generated for each simulation run for MIT Reality and UCSD trace, respectively, with the sources and destinations selected randomly from nodes within each trace. Different number of messages are used due to the fact that traces have different duration. The first half of the trace is used as a warmup period, when each protocol estimates the node quality metric they use. We set $q_{init} = 1$, $q_{cost} = 4 \times 10^{-5}$ for Reality and 2×10^{-4} for UCSD. q_{cost} is selected such that all nodes' memory load quota remain positive at the end of simulation.

Our simulator implements a similar energy module as TheONE [42], i.e., we consider both packet transmission and reception. Notice that our analysis in Section 5.2.2 has shown that energy consumption is proportional to the expected data size. It is therefore sufficient to consider the number of messages transmitted and received for a node as an indication of its energy consumption [79]. We omit devices' energy consumption for baseline activity and scanning, as we focus on the energy consumption incurred by routing protocols.

We use the message deadline as a simulation parameter, varying from 1 hour up to 24 hours. Since deadline is a typical parameter for evaluating routing performance, we explore its impact on energy consumption imbalance as well. We empirically set $\rho = 15$ for the Reality trace and $\rho = 4$ for the UCSD trace (ρ is selected such that it maintains comparable data copy overhead with other protocols while reduces energy consumption imbalance). Five simulations are run, with random seeds for statistical

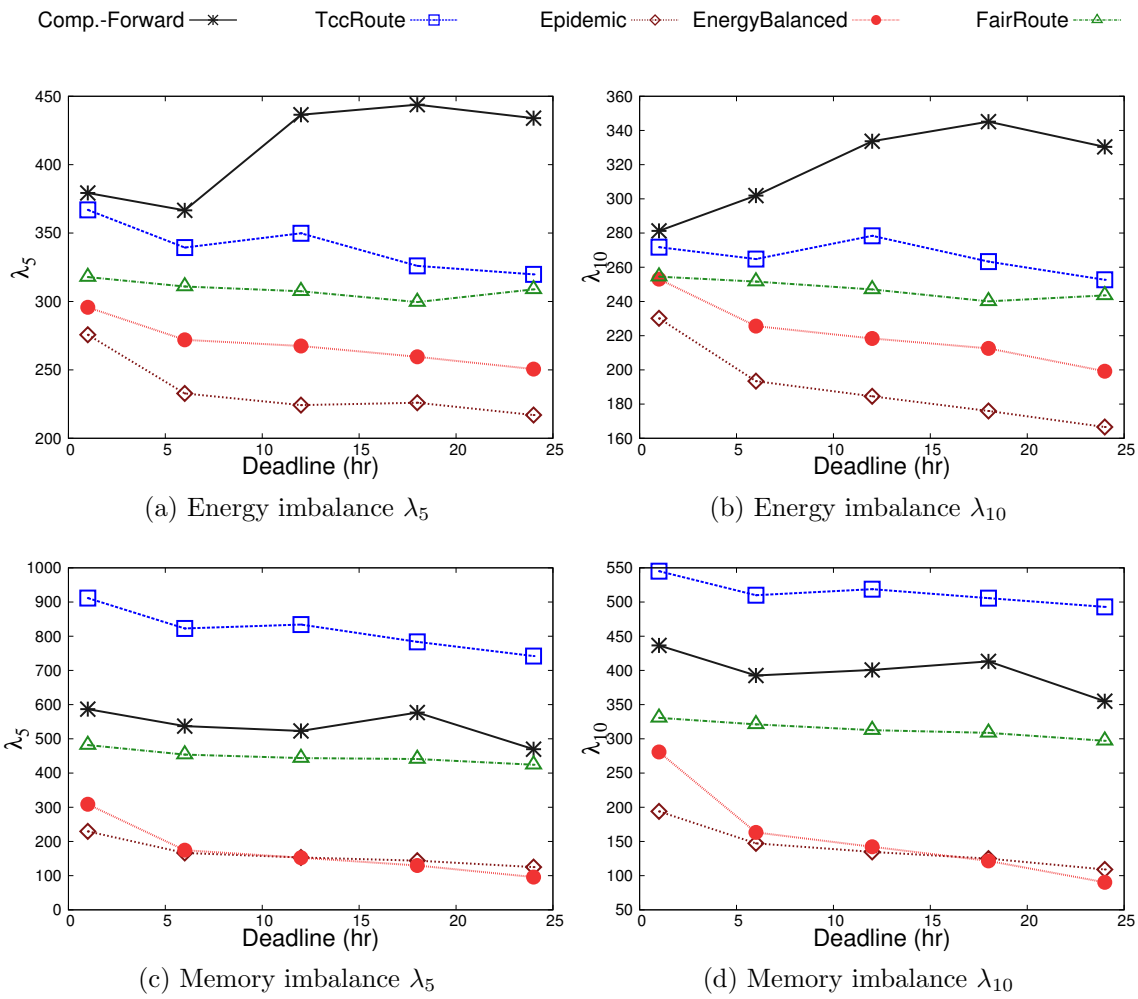
significance.

We compare our Energy Consumption Balanced Routing protocol with **Compare-Forward**, **TccRoute** [92], **FairRoute** [67], and **Epidemic** routing [82]. TccRoute is a state-of-art TCC-aware routing protocol that uses CCP as node quality metric. FairRoute is a routing protocol that achieves fairness among nodes in OMN. *Aggregated interaction strength* is used as quality metric, and a fairness mechanism based on queue length is proposed to prevent popular nodes from receiving excessive amount of traffic. A node forwards a message only if the encountered node has higher quality metric and smaller queue length. For fairness, we implement a TCC-aware FairRoute because [92] demonstrates that TCC-awareness improves routing performance, and because our protocol is also TCC-aware. Epidemic is essentially a flooding-based protocol, and is typically used as the baseline comparison for routing performance. In this section, we use it as baseline for energy consumption imbalance, since nodes' opportunity to be message carriers solely depends on mobility.

5.4.1 Evaluation Metrics

We evaluate the energy consumption imbalance, the memory load imbalance and routing performance. We use L_i to denote either energy consumption or memory load for v_i.

For evaluating energy imbalance, we use the **percent imbalance** metric. This metric is typically used to measure load imbalance in distributed processing systems [63]. Since these two problems are similar, this metric is suitable for both. The metric is defined as: $\lambda_k = \left(\frac{\tilde{L}_k}{\overline{L}} - 1\right) \times 100\%$, where $\tilde{L}_k$ is the average value of top $k\%$ L_i values and $\overline{L} = \frac{1}{N}\sum_i L_i$. *The load imbalance is more severe if the percent imbalance metric is high.* We show the results for $k = 5, 10$ for both energy consumption imbalance and memory load imbalance.



(a) Energy imbalance λ_5

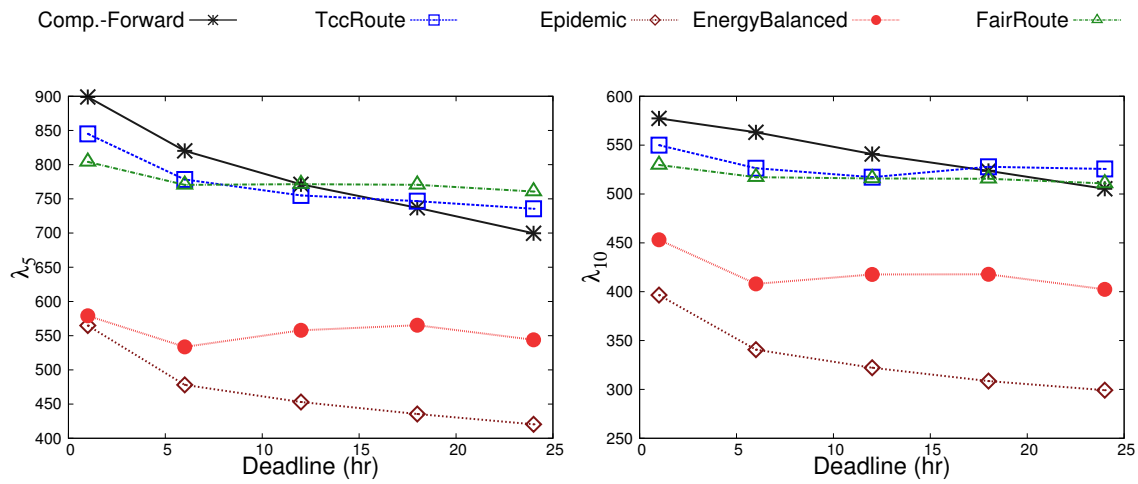
(b) Energy imbalance λ_{10}

(c) Memory imbalance λ_5

(d) Memory imbalance λ_{10}

Figure 5.6: Energy consumption and memory load imbalance for the MIT Reality trace

In addition to the percent imbalance metric, we also consider traditional routing performance metrics, including the **Packet Delivery Ratio** (PDR), the **Packet Delivery Delay** (PDD) and **Data Copy** (the number of copies created for each message, indicating the overhead of the routing protocol).



(a) Energy imbalance λ_5

(b) Energy imbalance λ_{10}

(c) Memory imbalance λ_5

(d) Memory imbalance λ_{10}

Figure 5.7: Energy consumption and memory load imbalance for the UCSD trace

5.4.2 Evaluation for Energy Consumption and Memory Load Imbalance

In this section we present the imbalance results, shown in Figure 5.6 for Reality and Figure 5.7 UCSD traces. The ultimate goal for our protocol is to achieve better energy consumption balance while maintaining routing performance. From Figures 5.6 and 5.7 we can see that, as expected, Epidemic achieves the best balance performance among all routing protocols, for both energy consumption and memory load balance. This is due to the fact that Epidemic floods messages to the entire

Deadline (hr)	Comp.&Forward	Epidemic	FairRoute	TccRoute
1	1.13	4.55	1.25	1.69
6	1.32	4.27	1.17	1.39
12	2.12	4.31	1.14	1.37
18	1.90	4.64	1.13	1.30
24	1.69	4.54	1.14	1.18

Table 5.3: Life time ratio results for the MIT Reality Trace

Deadline (hr)	Comp.&Forward	Epidemic	FairRoute	TccRoute
1	1.36	6.22	1.64	1.70
6	1.39	4.88	1.39	1.14
12	1.34	4.35	1.30	0.92
18	1.55	4.05	1.26	0.82
24	1.51	3.88	1.25	0.77

Table 5.4: Life time ratio results for the UCSD Trace

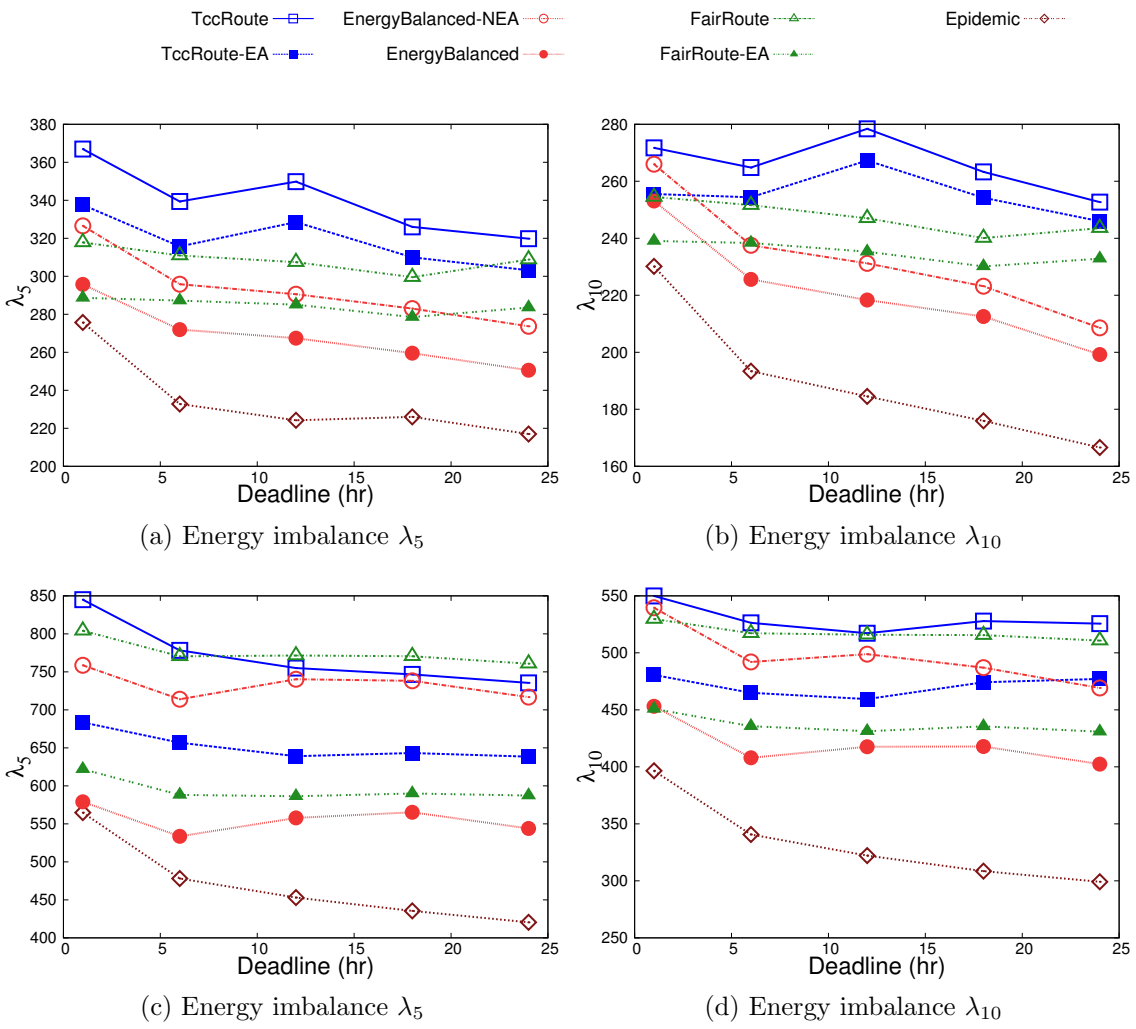
population of the nodes. From Figures 5.6a, 5.6b, 5.7a and 5.7b we can see that our protocol is the second best in terms of energy consumption balance (only worse than Epidemic) for all deadlines. Compared to the FairRoute protocol, our protocol achieves up to 19% and 18% reduction of energy consumption imbalance in the Reality trace for λ_5 and λ_{10}, respectively; while in UCSD, our protocol achieves up to 31% and 21% reduction for λ_5 and λ_{10}, respectively. This result shows that our protocol reduces the energy consumption imbalance.

On the other hand, we notice that our protocol performs significantly better for memory load imbalance reduction, as shown in Figures 5.6c, 5.6d, 5.7c and 5.7d. This result is expected, as we explicitly minimize the memory load imbalance.

To show how the proposed protocol prolongs the lifetime of the mobile nodes, we assume that the energy consumption rate is proportional to the total energy consumption. Consequently, assuming that all the nodes have identical initial energy

level, the lifetime of the mobile nodes is proportional to the reverse of the total energy consumption. Denote T_R as the *time until the first node dies when running routing protocol R*. Table 5.3 and 5.4 show the ratio of T_{EBR}/T_R, where EBR stands for Energy Balanced Routing, and $R \in$ {Compare-and-forward, Epidemic, FairRoute, TccRoute}. As we can see, in general, EBR extends the lifetime by 13% to 2x when compared to social-based routing protocols, i.e., Compare-and-forward, FairRoute and TccRoute, through balancing the energy consumption. EBR extends the lifetime by 3.88x to 6.22x when compared to Epidemic through reducing packet replication. However, we also notice that EBR has shorter lifetime when compared to TccRoute for large deadlines. A closer look at the routing performance reveals that TccRoute has fewer packet replication, which saves a lot of energy.

As our *Energy-aware Intra-TCC routing* (Section 5.3.4) is independent of the OMN forwarding decision, it is interesting to see how it might impact other TCC-aware routing protocols. To this end, we implement TccRoute and FairRoute with *Energy-aware Intra-TCC routing*, which are then named as TccRoute-EA and FairRoute-EA. We also implement our protocol without the *Energy-aware Intra-TCC routing*, named as EnergyBalanced-NEA. Since intra-TCC routing does not affect memory load, we only present the results for energy imbalance, as are shown in Figure 5.8. First, we note that even EnergyBalanced-NEA outperforms TccRoute and FairRoute in most cases, demonstrating that our memory load controlling mechanism is effective. Second, all TCC-aware routing protocol consistently benefit from *Energy-aware Intra-TCC routing*, as we can see that the imbalance is reduced for all deadlines. Finally, we note that our protocol still outperforms TccRoute-EA and FairRoute-EA in most cases.



(a) Energy imbalance λ_5

(b) Energy imbalance λ_{10}

(c) Energy imbalance λ_5

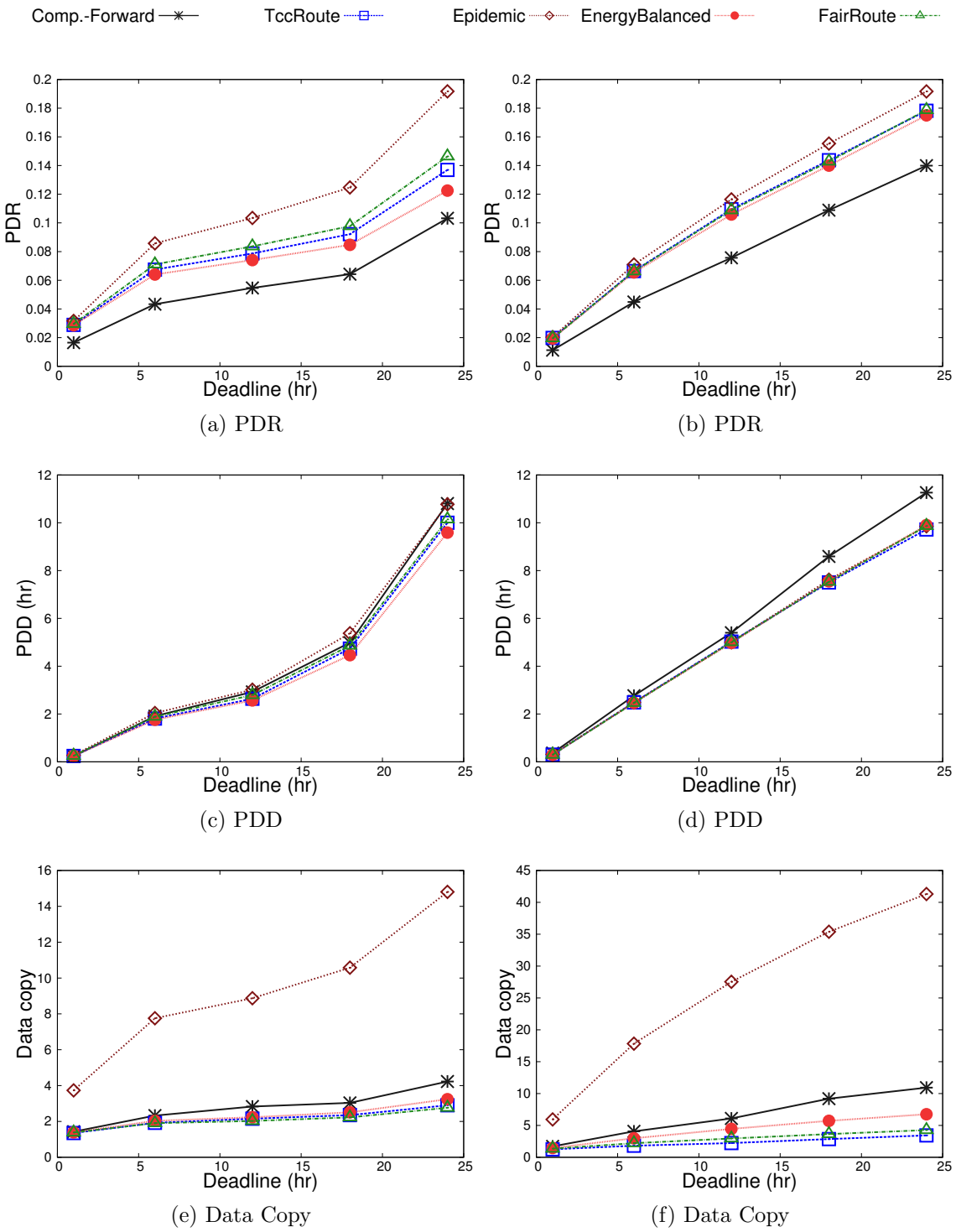
(d) Energy imbalance λ_{10}

Figure 5.8: Applying the Energy-aware intra-TCC routing technique to other routing protocols: (a) (b) for the MIT Reality trace; (c) (d) for the UCSD trace

5.4.3 Evaluation for Routing Performance

In this section we present the routing performance results as shown in Figure 5.9. Notice that we achieve comparable routing performance with other TCC-aware social-based routing protocols, in terms of PDR, PDD and Data copy. PDR is slightly decreased in Reality, while it is comparable in UCSD. Our protocol achieves the same PDD as other protocols for both traces. As for data copy overhead, our

protocol has comparable overhead for Reality while slightly increased overhead for UCSD. These results demonstrate a trade-off between PDR, data copy and the energy imbalance. Reducing imbalance results in decreased PDR when maintaining overhead, or increased overhead when the PDR is maintained.



(a) PDR (b) PDR

(c) PDD (d) PDD

(e) Data Copy (f) Data Copy

Figure 5.9: Routing performance: (a), (c) and (e) for the MIT Reality trace; (b), (d) and (f) for the UCSD trace.

6. HYBRID ROUTING PROTOCOL[1]

Real world deployments of wireless networks often exhibit diverse connectivity characteristics. In a wireless network deployed in developing areas [19], factors such as interference, unreliable power, etc. resulted in unreliable wireless links, which leads to a network with rather dynamic connectivity. In ad hoc networks deployed in disaster response scenarios, device carriers (e.g., first responders, patrol cars, ambulances, etc) are highly mobile. Consequently, the network connectivity often ranges from well connected to almost disconnected [13] [70]. In opportunistic social networks, application (e.g., 1am [50], Firechat [16]) users may form well connected networks with time-varying topology [92], despite the fact that they are disconnected most of the time due to their different schedules. In a vehicular delay tolerant network (DTN) in which vehicles move in an area where multiple wireless mesh networks are also deployed [80], a hybrid network is formed with spatially diverse connectivity.

Given the proliferation of wireless capable devices, users have increasing opportunity to encounter networks with diverse connectivity in the future. However, routing protocols that are designed for one type of connectivity (e.g., well connected wireless mesh network) perform poorly for other types (e.g., intermittently connected DTNs). Therefore, the design of a hybrid routing protocol that can adapt itself in uncertain environments and that can perform well in the entire range of the connectivity characteristics is of critical importance to the guarantee of routing performance.

Tie et al. [80] has identified that the key structural difference between routing protocols for connected networks and for DTNs is packet replication. Their pro-

[1]Part of this section is reprinted from "Hybrid routing in wireless networks with diverse connectivity" by Chen Yang, Radu Stoleru, in Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '16). ACM, New York, NY, USA, 71-80. DOI 10.1145/2942358.2942374

posed R3 routing protocol dynamically replicates packets on multiple paths. R3 uses a model for the estimation of the replication gain, i.e., the delay reduction of a packet if multiple copies are sent simultaneously through paths whose delays are assumed to be independent. R3's dynamic replication approach is in sharp contrast to previous solutions, which either adopt a single-copy approach [19] [70] [18] (only one copy of the packet is sent) that fail to deliver satisfactory performance in DTNs, or empirically set the replication according to their experiment scenarios [44] [84], which might not work well in other environments.

Although R3 makes an important step towards the successful design of hybrid routing protocol, there are multiple limitations. First, we observe that the *correlation* between path delays can significantly and adversely affect the delay reduction obtained from replications. By assuming independent path delays, one may overestimate the replication gain for a set of paths, and thus underestimates how many copies should be created for a given packet in a specific network environment. Such a suboptimal decision may lead to decreased packet delivery ratio and increased packet delays in highly dynamic networks. Moreover, the use of source routing when distributing packet copies significantly limits the routing performance in highly dynamic DTNs. It has been shown that source routing is inefficient in DTN environments [40] since intermediate nodes cannot forward the packet to better carriers which are met after the routing decisions have been made at the source. Furthermore, the computational cost for selecting additional paths grows exponentially when trying to support more replications, which can be inefficient in large scale networks.

In this section, we present the design of a hybrid routing protocol that performs well in the entire range of connectivity characteristics. To approach this problem, we carefully study the benefit of replication in terms of delay reduction, as it is the key to understand *when and how much replication should be used*. To accurately

estimate the benefit of replication, we propose a novel model to capture the potential correlation between inter-contact times for different nodes. We then propose a regret-minimization based algorithm to dynamically decide how much replication should be used under the current network environment, taking into account the potential delay correlations that might impact the benefit of replication. Enabled by these two important tools, we design the Hybrid Routing Protocol (HRP) that dynamically adjusts the amount of replication and efficiently forwards packets under any network conditions. We further implement HRP as a user space daemon program and deploy it to a testbed which consists of nine wireless routers. Through extensive simulations in networks with diverse connectivities and real world evaluations in the wireless testbed, we demonstrate that HRP outperforms state-of-art hybrid routing protocols and achieves comparable performance with specialized protocols for a given network environment.

The contributions of this section are as follows:

- We demonstrate the impact of path delay correlation on the benefit of replication, and propose a novel model for estimating inter-contact time without loosing the correlation information. Such a model allows for accurate analysis of replication and the design for efficient forwarding strategies.

- We propose a novel regret minimization based algorithm which adaptively decides the appropriate amount of replication given any network environment.

- We design and implement the Hybrid Routing Protocol which adopts a novel forwarding metric called *Additional Contact Rate* (ACR). We demonstrate through extensive simulations that HRP outperforms state-of-art hybrid routing protocols by up to 3x in terms of delay improvement with comparable, if not lower, overhead.

- We deploy HRP to a wireless testbed consisted of nine Mikrotik wireless routers and conduct experiments under different scenarios. We show that HRP achieves competitive performance when compared to specialized routing protocols for well connected and sparsely connected networks.

The organization of this section is as follows. We present the impact of delay correlation on replication benefit in Section 6.1, followed by the presentation of the joint Inter-Contact Time model. In Section 6.2, we show the regret minimization based algorithm for deciding replication factors. We present the design of HRP in Section 6.3. In Section 6.4 we show the results of extensive simulations, followed by the implementation details and real world experiment results in Section 6.5.

6.1 Modeling Inter-contact Time Correlation

In this section we propose a novel model for Inter-Contact Time (ICT) estimation that incorporates the correlation among a group of nodes. Inter-contact time refers to the time duration between two contact events between a pair of nodes. It is the main source of routing delay in sparsely connected networks [80] and is an important metric for measuring node's delivery capability [4]. Through a simple example, we show that delay correlations among different nodes can significantly affect the benefit of replication. Motivated by this observation, we propose a novel Joint ICT model that quantitatively captures the correlation among different nodes. Our model allows us to estimate the replication benefit more accurately and thus facilitates the design of a more efficient routing protocol.

6.1.1 Preliminary

We first introduce two important concepts that are used extensively throughout the section, namely *replication factor* and *replication gain*.

Definition 7. *Replication factor r: the total number of data copies created at the source for a given packet.*

Replication factor limits the total number of copies for a packet, and is used in many quota-based DTN routing protocols, such as [79].

We define *replication gain* for a routing protocol, originally proposed in [80], as follows: let D_r be the random variable for routing delay when *replication factor* is r, then:

Definition 8. *Replication gain* $\gamma_r = E[D_1]/E[D_r]$.

Replication gain captures the benefit of replication in terms of delay improvement. Notice that γ_r depends on the forwarding strategy of a routing protocol. In particular, if the packet copies are routed along a set of paths P, where each path has delay X_i, then the replication gain is $\gamma_r^P = \min_{i\in P}\{E[X_i]\}/E[\min_{i\in P}\{X_i\}]$.

Table 6.1: Mathematical symbols used in Section 6

r	Replication factor
γ_r	Replication gain for r
$\boldsymbol{\pi}$, π_r	Probability distribution for replication factors
$\boldsymbol{s}$, s_i	Vector in $\{0,1\}^N$, representing a group of nodes
$\lambda_{\boldsymbol{s}}$	Contact rates for a group of nodes $\boldsymbol{s}$

We summarize the mathematical symbols in Table 6.1.

6.1.2 Delay Correlation Impacts Replication Gain

To illustrate how *delay correlation adversely affects the benefit of replication*, consider a simple example as shown in Figure 6.1: node S sends packets to D with replication factor $r = 2$. S can use paths $\{X_i\}_{i=1}^3$, consisting of links $\{Y_j\}_{j=1}^4$, which

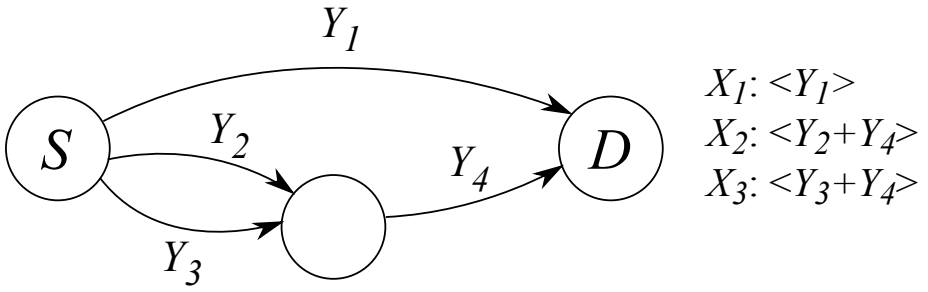


Figure 6.1: A simple network model where $\{Y_j\}_{j=1}^4$ denotes link delays and $\{X_i\}_{i=1}^3$ denotes path delays.

are independently and exponentially distributed. Then the replication gain from using paths $\{X_i, X_j\}$ is $\gamma_2^{i,j} = \frac{\min\{E[X_1],E[X_2],E[X_3]\}}{E[\min\{X_i,X_j\}]}$, since when $r = 1$ the shortest path is used. Notice that the delays of X_2 and X_3 are positively correlated due to the common link Y_4, and the correlation becomes stronger when $E[Y_4]$ increases. X_1 is independent of both X_2 and X_3.

When all paths have identical expected delay, correlated paths always have smaller replication gain, and the replication gain decreases when the correlation becomes stronger. We fix $E[X_i] = 60$ for $i = 1, 2, 3$ and vary $E[Y_4]$ while examine the replication gain $\gamma_2^{1,2}$ and $\gamma_2^{2,3}$. As shown in Figure 6.2, $\gamma_2^{1,2}$ remains relatively high due to the independence of delay. But $\gamma_2^{2,3}$ quickly approaches 1 when $E[Y_4]$ increases, indicating a drastic decrease of benefit from replication. If S assumes independence of path delays, it cannot accurately estimate the replication gain and may make suboptimal decisions. Indeed, *redundancy helps improve system performance, but we should utilize resources as diverse as possible to maximize the benefit* [83].

However, *if independent paths have significantly inferior performance, it might still be wise to utilize correlated paths.* To see this, we fix $E[X_2] = E[X_3] = 60$ and let $E[X_1] = 60 + x$ where $x > 0$. By varying both x and $E[Y_4]$, we observe that there exist threshold x^*, above which $\gamma_2^{2,3}$ becomes larger than $\gamma_2^{1,2}$ as shown in Figure 6.3.

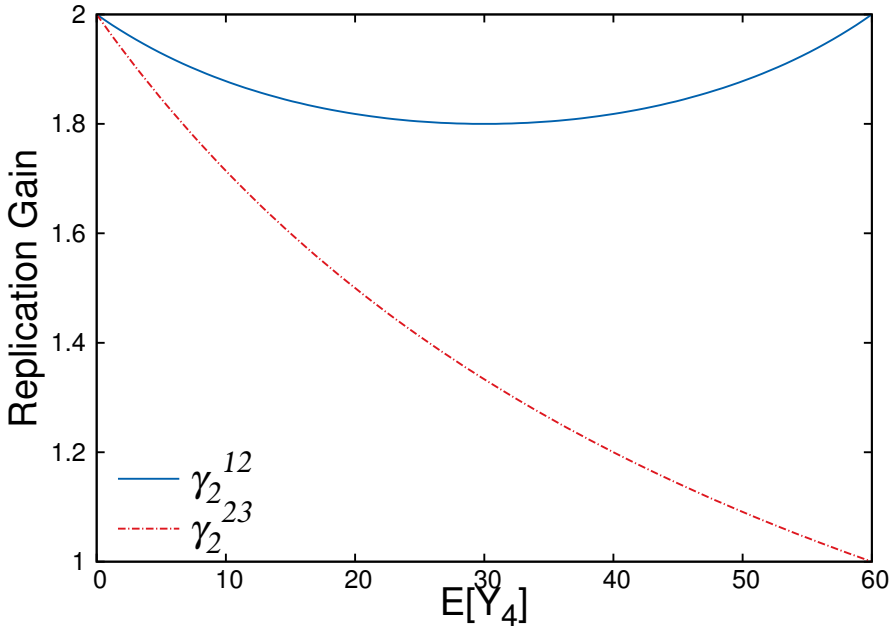


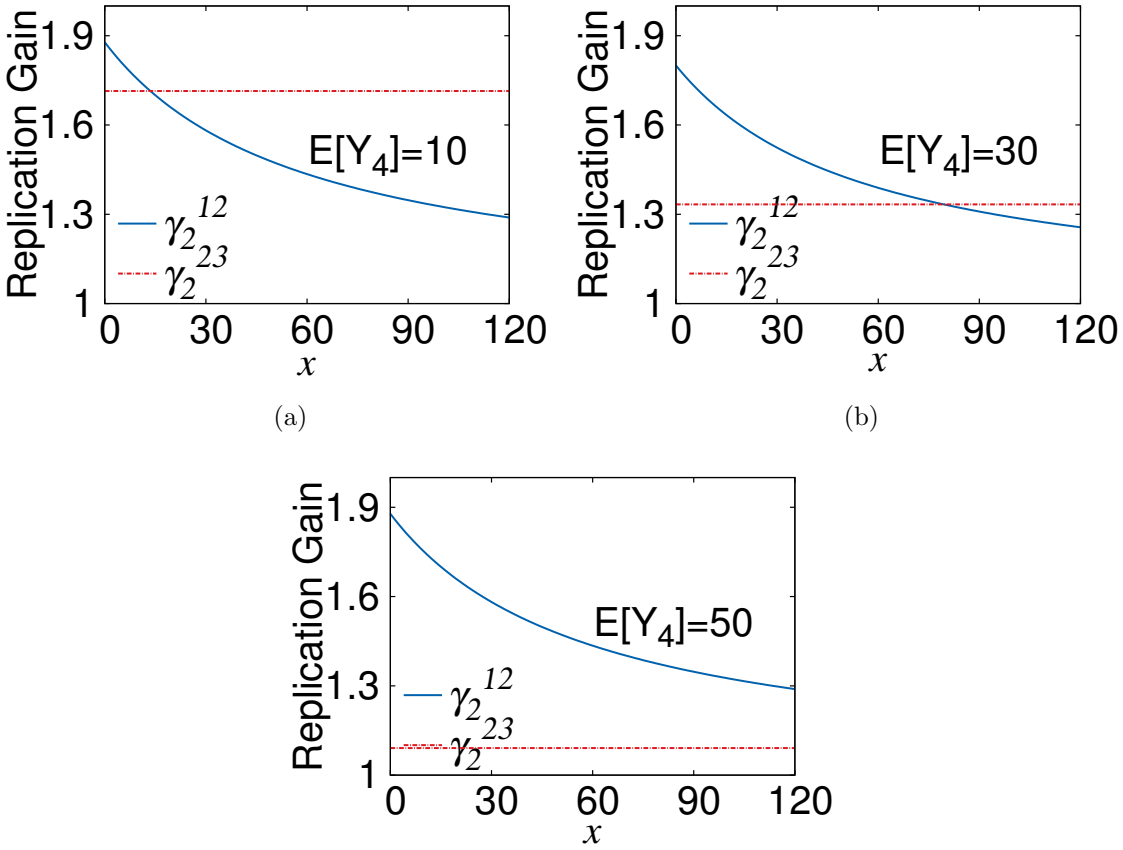
Figure 6.2: Fix $E[X_i] = 60$ min, $i = 1, 2, 3$, and vary $E[Y_4]$.

This indicates that better performance is achieved when correlated paths are used. Notice that x^* increases with $E[Y_4]$. It is related to the correlation between X_2 and X_3.

Therefore, *in order to accurately estimate the replication gain and make better routing decision, it is important to capture the delay correlation.*

6.1.3 Joint Inter-contact Time Distribution

In sparsely connected networks where replication is widely adopted, ICT is the main source of link delay [80], and is also an important metric for measuring delivery capability [4]. Given the insights from the previous section, we propose to jointly capture the ICTs between nodes such that the correlation information is preserved, which enables more accurate estimation of the replication gain.



(a) (b)

(c)

Figure 6.3: Fix $E[X_2] = E[X_3] = 60$ min, and vary x.

6.1.3.1 2-dimensional Joint ICT Distribution

Intuitively, the ICTs of two nodes are correlated if the observer often meets both of them at the same time. Consider the contact processes between node v, and nodes v_1 and v_2, as shown in Figure 6.4 (the right hand side of which shows the contact processes). Notice that node v has met v_1 and v_2 for 7 and 6 times, respectively. Out of these, 4 times it meets them at the same time. In this case, ICTs X_1 and X_2 can no longer be assumed as independent.

We use the *Marshall and Olkin's Bivariate Exponential Distribution* (BVE) [54]

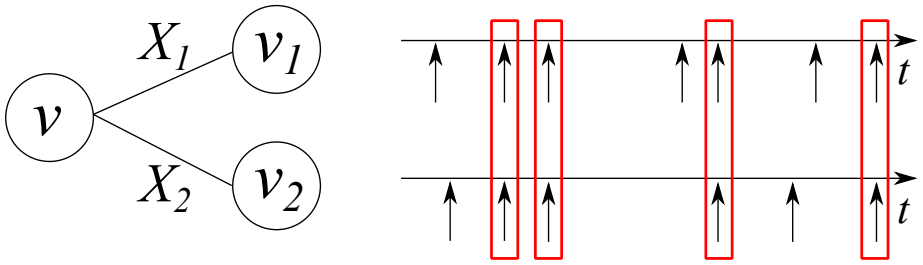


Figure 6.4: Contact process between (v, v_1) and (v, v_2).

to model the 2-D joint ICT distribution for a pair of nodes.

Definition 9. *Marshall and Olkin's Bivariate Exponential Distribution: random variables X_1 and X_2 follow BVE($\lambda_1, \lambda_2, \lambda_{12}$), if the joint distribution satisfies $Pr(X_1 > x_1, X_2 > x_2) = e^{-(\lambda_1 x_1 + \lambda_2 x_2 + \lambda_{12} \max(x_1, x_2))}$*

BVE was derived from a "fatal shock" model of a two component system, which is similar to the contact process shown in Figure 6.4. In the "fatal shock" model, shocks arrive at the system according to three independent Poisson processes with parameter λ_1, λ_2 and λ_{12}, and are applied to component 1, component 2, and both, respectively. The lifetimes of the two components X_1 and X_2 follow BVE. Similarly, the contact process between node v, and node v_1 and v_2, can be seen as three independent Poisson processes with parameters λ_1, λ_2 and λ_{12}, where two processes govern node v_1 and v_2 individually, while the third process characterizes the events when both v_1 and v_2 meet v. In this case, the inter-contact times X_1 and X_2 are joint distributed as a BVE. Notice that the marginals are still exponential, and the correlation coefficient between X_1 and X_2 is $\lambda_{12}/(\lambda_1 + \lambda_2 + \lambda_{12})$.

Maintaining exponential marginal distribution for ICTs is important, as we notice that exponential distributions can be used to characterize ICTs in many mobile scenarios (e.g., Vehicular Adhoc Networks [93], Opportunistic Social Networks [29] [62]).

6.1.3.2 Multi-dimensional Joint ICT Distribution

Although BVE can be used to model correlated inter-contact time for a pair of nodes, in real world scenarios, a node may also meet with a *group of nodes* on a regular basis. It is therefore also necessary to characterize the joint distribution of a group of nodes. Notice that a BVE is a special case of a *Multivariate Exponential Distribution* (MVE) [54]. We can therefore use MVE to model multi-dimensional joint ICT. MVE is defined as follows: Let S denote the set of vectors $\boldsymbol{s} = (s_1, \ldots, s_n)$ where each $s_i = 0$ or 1, but $(s_1, \ldots, s_n) \neq (0, \ldots, 0)$. For a vector $\boldsymbol{s} \in S$, let $\max(x_i s_i)$ denote the maximum of the x_i's for which $s_i = 1$.

Definition 10. *Multivariate Exponential Distribution: Random variables $X_1, \ldots, X_n$ follow MVE, if their joint distribution satisfies:*
$Pr(X_1 > x_1, \ldots, X_n > x_n) = \exp\left[-\Sigma_{\boldsymbol{s} \in S} \lambda_{\boldsymbol{s}} \max(x_i s_i)\right]$

As an example, consider $n = 3$: $\lambda_{(100)} = \lambda_1$, $\lambda_{(010)} = \lambda_2$, $\lambda_{(001)} = \lambda_3$ are parameters for Poisson processes that govern each individual node; $\lambda_{(110)} = \lambda_{12}$, $\lambda_{(101)} = \lambda_{13}$, $\lambda_{(011)} = \lambda_{23}$, are parameters that govern each pair of nodes; and $\lambda_{(111)} = \lambda_{123}$ is the parameter that governs all three nodes. Marginal distribution of X_1 and X_2 follows a BVE$(\lambda_1', \lambda_2', \lambda_{12}')$, where $\lambda_1' = \lambda_1 + \lambda_{13}$, $\lambda_2' = \lambda_2 + \lambda_{23}$, $\lambda_{12}' = \lambda_{12} + \lambda_{123}$.

Throughout this section, we also use $\boldsymbol{s}$ to denote a group of nodes, i.e., the node set $\{v_i : s_i = 1\}$.

6.1.4 MVE Parameter Estimation

In a real world scenario, a node experiences a sequence of contact events, from which it estimates the parameters of the MVE distribution. We adopt a simple *consistent* and *unbiased* estimator derived in [3].

Consider that a node v has N encounter events within time T. At each event node

Algorithm 4 MVE Parameter Estimation For Node v

```
1: Initialize EncounterSet = ∅, T = 0, N = 0
2: nextTimer ← δ
3: upon connection up event
4: Add newneighbor to EncounterSet
5:
6: upon nextTimer fired event
7: Set nextTimer ← nextTimer + δ
8: T = T + δ
9: if EncounterSet ≠ ∅ then
10:     N = N + 1
11:     if no counter exist for EncounterSet then
12:         Create new counter for EncounterSet
13:         counter = 0
14:     end if
15:     Increment counter for EncounterSet
16:     EncounterSet ← ∅
17: end if
18: Update parameters for all EncounterSet according to Equation 6.1
```

v may meet one or more nodes. Let $Y_{\boldsymbol{s},j} = 1$ if at the j-th event node v encounters a set of nodes $\{v_i : s_i = 1\}$; and $Y_{\boldsymbol{s},j} = 0$ otherwise. Then the estimation of $\lambda_{\boldsymbol{s}}$ given these N observations is given by:

$$\hat{\lambda}_{\boldsymbol{s},N} = \frac{\frac{1}{N}\Sigma_{j=1}^{N} Y_{\boldsymbol{s},j}}{\frac{T}{N-1}} \tag{6.1}$$

In the example shown in Figure 6.4, there are $N = 9$ contact events for node v. Assuming that the last event is at time T, according to Equation 6.1, we have $\hat{\lambda}_{(10),9} = \frac{8}{3T}$, $\hat{\lambda}_{(01),9} = \frac{16}{9T}$, $\hat{\lambda}_{(11),9} = \frac{32}{9T}$. In real world scenarios, contact events with multiple nodes do not happen at the exact same time. We therefore aggregate multiple contact events that happen within a *estimation time window* as a single contact event with multiple nodes.

We present the MVE parameter estimation algorithm for a node v in Algorithm 4.

We first initialize T, N and *EncounterSet* (which is used to aggregate nodes contacted in the estimation time window δ). The *nextTimer* is scheduled to fire at time $t = \delta$. Upon each contact event, the newly encountered node is added to *EncounterSet* (lines 3 and 4). When the *nextTimer* is fired, we update parameters. First, the *nextTimer* is set up (line 7), and T is incremented by one estimation time window. If the *EncounterSet* is not empty, node v has met some nodes during the time window. The total contact event counter N is updated (line 10). Next, if no *counter* exists for the encountered set of nodes, a new counter is created (lines 11 to 14). Then the *counter* for the encountered set of nodes is incremented, and the *EncounterSet* is reset to empty. Finally, all the parameters are updated using Equation 6.1.

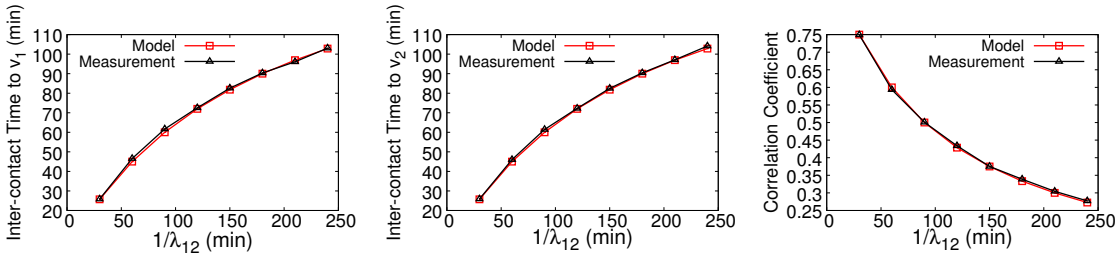
According to [3], the estimator which we adopt is *consistent*, i.e., the parameter estimation ultimately converges to the ground truth. The consistency follows directly from the formula of mean squared error (m.s.e.) of $\hat{\boldsymbol{\lambda}}_N$ from $\boldsymbol{\lambda}$ (see [3]):

$$m.s.e.(\hat{\boldsymbol{\lambda}}_N) = [(N-1)\lambda^2 + \sum \lambda_{\boldsymbol{s}}^2]/[N(N-2)]$$

where $\hat{\boldsymbol{\lambda}}_N$ is the vector of estimated parameters, $\boldsymbol{\lambda}$ is the ground truth, and $\lambda = \sum_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}$. As N increases, the m.s.e. drops to zero. In other word, with more samples collected from the contacts, the estimation accuracy would keep increasing. The time of getting an accurate estimate depends on the number of contacts as well as the inter contact time durations.

6.1.5 BVE Validation Using Poisson Processes

To validate the proposed BVE model, we generate contact processes using Poisson processes. To this end, assuming node v meets with node v_1 and v_2. We generate Poisson processes with parameters λ_1, λ_2 and λ_{12}, respectively, to simulate the con-



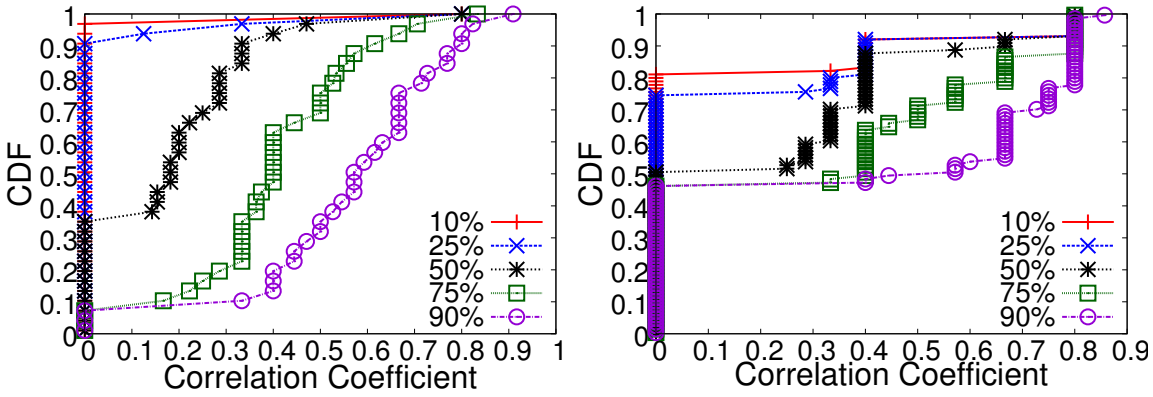
(a) Inter contact times to node v_1 (b) Inter contact times to node v_2 (c) Correlation coefficient

Figure 6.5: BVE model validation

tacts between v and v_1, v and v_2, and v and both v_1 and v_2, respectively. We generate a one-week-long contact trace, and run the proposed algorithm on node v. Figure 6.5 shows the results of Inter-contact Times to v_1 (ICT_1) and v_2 (ICT_2), as well as the correlation coefficient (r), while varying $1/\lambda_{12}$, i.e., the inter-arrival time of contacts between v and both v_1 and v_2. A smaller $1/\lambda_{12}$ indicates larger correlation between the ICT_1 and ICT_2. We compare the measurement results to the model predictions, i.e., $ICT_1 = 1/(\lambda_1 + \lambda_{12})$, $ICT_2 = 1/(\lambda_2 + \lambda_{12})$, and $r = \lambda_{12}/(\lambda_1 + \lambda_2 + \lambda_{12})$. As we can see from the figures, the measurement results fit perfectly with the model.

Table 6.2: Mobility traces used for the analysis and evaluation of the Hybrid Routing Protocol

Trace	*MIT Reality*	*UCSD*
Device	Phone	PDA
Network type	Bluetooth	WiFi
No. devices	97	275
Duration(days)	246	77



(a) Reality (b) UCSD

Figure 6.6: Correlation coefficient for pairs of nodes

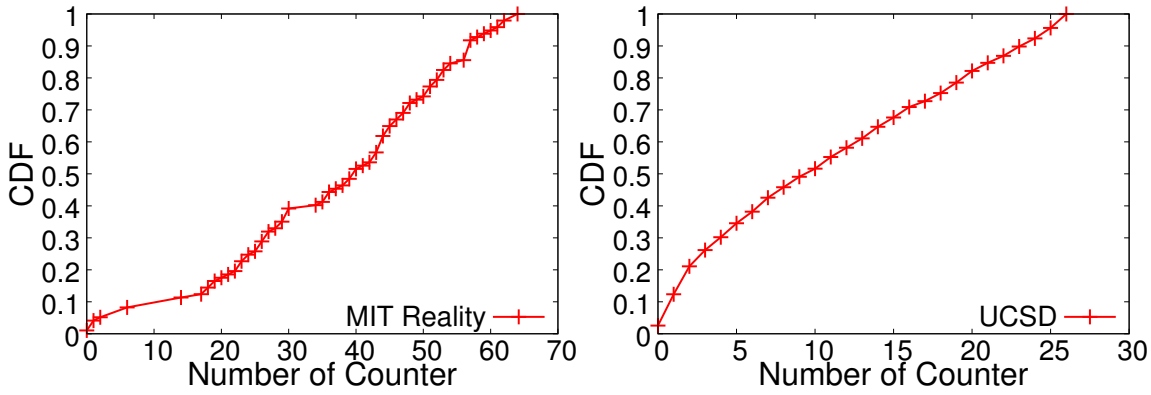
6.1.6 ICT Correlation in Real World Mobility Traces

We use two real world mobility traces to validate the ICT correlation. The details of the datasets are shown in Table 6.2. As shown, the *Reality* [22] trace consists of 97 users, selected from MIT students, staff and faculty members. The mobility trace contains Bluetooth connection events over a time period of nine months. *UCSD* [55] records the Access Point detection and association events for 275 hand-held PDAs distributed to students. The trace covers eleven weeks. We assume that there is a contact event between two users if the time windows when they detect the same AP, overlap.

We replay each mobility trace and have each node v run the Algorithm 4 throughout the entire trace and collect the MVE parameters.

6.1.6.1 ICT Correlation

Notice that the MVE parameter collected by a node essentially represents a multi-dimensional distribution, where each dimension represents an encountered neighbor. Therefore, for each pair of the neighbors a node encounters, we can derive a 2-D



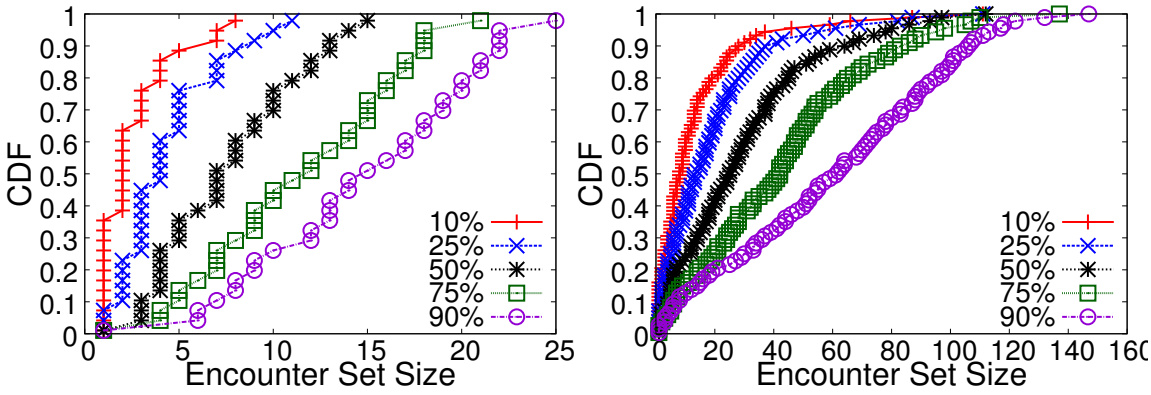
(a) Reality (b) UCSD

Figure 6.7: Distributions of the number of EncounterSet

marginal distribution from the estimated parameters, which allows us to measure the correlation coefficient ρ between ICTs to each of the two nodes. Then at each node, we can get an empirical CDF of ρ values calculated from each pair of the neighbors.

To provide a more comprehensive view of all the correlation coefficients, we extract the percentile values of the aforementioned CDFs from each node, and plot the CDF of these values in Figure 6.6. For example, a point of $(0.3, 0.7)$ on the 50% line indicates that for 70% of the nodes, the median ρ among all the pairs it has met is less than or equal to 0.3. As we can see from Figure 6.6, the median of ρ is up to 0.7 (Reality) and 0.8 (UCSD). From each node's view, a significant proportion of pairs of nodes it has met have correlated ICT and the correlations are strong. *The independence assumption of node mobility thus does not hold.*

Figure 6.7 shows the distribution of the number of *EncounterSet*s each node meets at the end of the trace. Notice that each node only needs to maintain a reasonable number of *EncounterSet*s to estimate the MVE parameters. This is due to the fact that the mobility of nodes is heterogeneous and each node only meets a small number



(a) Reality (b) UCSD

Figure 6.8: Distributions of the sizes of the encounter node set

of groups of nodes frequently.

In Figure 6.8, we use the same approach as presented above and show the CDFs of different percentile values of the *EncounterSet* size. We note here that nodes typically meet with multiple nodes at a time.

6.2 Regret Minimization Based Algorithm for Dynamically Choosing Replication Factor

Now that we have seen how delay correlation can affect replication gain and how to capture potential ICT correlations, we still need a mechanism to decide appropriate replication factors. In this section, we present a novel algorithm based on regret minimization [60] to choose replication factor dynamically, without the assumption of independent delays as it may bias the estimation of replication gain. The basic idea is to use a probability distribution to represent the preference of choosing a particular replication factor. Routing protocols then draw replication factors using the probability distribution. Our solution thus allows for seamless transition between a single-copy protocol for well connected mesh and a multi-copy protocol for sparsely

connected DTNs.

6.2.1 Basic Idea

Our basic idea is to consider the node as a player, who needs to repeatedly take actions (choosing a replication factor for a packet) under an *uncertain* environment (unknown network condition). Each time the node takes an action, the environment (network) gives a feedback (e.g., delay, resource utilization, etc.). The goal of our algorithm is to learn the best policy, which is a *probability distribution* over a set of actions, for such an environment. A powerful technique for analyzing this type of problem is known as Regret Analysis [60]. We would like an algorithm which performs as good as the best action we could have used if we knew the network condition *a priori*. The performance degradation between our algorithm and the best action is defined as *regret*.

6.2.2 Model and Problem Formulation

Let us consider a network with a set of nodes V and a pair of source and destination nodes (v_s, v_d). At time slot t, v_s chooses a probability distribution over a set of replication factors, i.e., $\boldsymbol{\pi}^t = \{\pi_i^t\}_{i=1}^{r_{max}}$. It then calculates a *loss vector* $l^t = \{l_r^t\}_{r=1}^{r_{max}}$ (defined in the following section) and experiences an expected loss of $\sum_{r=1}^{r_{max}} l_r^t \pi_r^t$. After T time steps, node v's total loss is $\sum_{t=1}^{T} \sum_{r=1}^{r_{max}} l_r^t \pi_r^t$.

Node v_s draws replication factors using the same distribution, and experiences the same expected loss. In retrospect, however, v_s could have chosen the same action i that results in the minimum total loss, i.e., with loss $L_i^T = \min_j \sum_{t=1}^{T} l_j^t$. The performance degradation of $\sum_{t=1}^{T} \sum_{r=1}^{r_{max}} l_r^t \pi_r^t - L_i^T$ is then defined as *regret*, which is to be minimized.

6.2.3 The Loss Function

We define the loss function to reflect the trade-off of two objectives, i.e., minimizing delay and resource utilization.

Definition 11. *The loss function for replication factor r:*

$$l_r^t = \alpha \cdot \frac{1}{\gamma_r} + (1-\alpha) \cdot \frac{r-1}{r_{max}-1}$$

where $\alpha \in [0,1], r_{max} > 2$.

Notice that $\frac{1}{\gamma_r} \in (0,1]$ is the inverse of replication gain, which approaches 1 when there is no benefit of replication. $\frac{r-1}{r_{max}-1}$ represents the loss of resource utilization. $\alpha \in [0,1]$ is a parameter that reflects the preference between optimizing performance and reducing overhead.

6.2.4 Computing the Replication Factor Distribution

To decide the probability distribution of replication factors, we adopt the *Polynomial Weighted* (PW) algorithm as it achieves near optimal performance in terms of regret minimization [60], which is significantly superior to other naive algorithms. We briefly introduce the PW algorithm as follows: each replication factor is assigned a weight w_r, initialized with value 1. At each time slot t, the node calculates a loss vector $\{l_r^t\}_{r=1}^{r_{max}}$. The weight w_r is updated according to: $w_r = w_r \cdot (1 - \eta \cdot l_r^t)$, where $\eta \in (0,1)$ is a constant. Then the probability of π_r is updated by $\pi_r = \frac{w_r}{\sum_i w_i}$.

If at each time slot the node can only calculates the loss value l_r^t for its chosen action r, instead of the entire loss vector, it is called a *Partial Information* model (PI), in contrast to the *Full Information* model (FI). PW can be generalized to solve PI problem, by partitioning time slots into blocks. The same probability vector is used within each block except for the exploration steps, when each replication factor

is chosen once. The loss received during each exploration step constitutes the loss vector and is used by the PI PW algorithm to update probability at the end of each block.

The performance of the Polynomial Weighted (PW) algorithm is near optimal [60]. It was shown that the worst-case regret achieved by the PW algorithm is upper bounded by $2\sqrt{T \log N}$, where the T is the time and N is the number of available actions. It was also shown that the optimal solution has a lower bound of $\Omega(\sqrt{T \log N})$, when $T \geq N$, as there exists a stochastic generation of losses that results in the lower bound regret.

In order to calculate the replication factor distribution we need to estimate the delay of packets under different replication factors. We present the detail of delay estimation in the following section.

6.3 Hybrid Routing Protocol

In this section, we present the design of our Hybrid Routing Protocol (HRP).

6.3.1 Overview

As shown in Figure 6.9, the core of HRP mainly consists of three components as presented in the shaded area. In addition, we assume that each node contains a Topology Managerment component that is able to broadcast and collect beacons in order to maintain the topology information for the network it currently connects to. It outputs events such as node connection and disconnection, which are used to trigger certain behaviors in HRP.

The outputted node events are then fed into the *MVE Estimation* module, which estimates MVE parameters using Algorithm 4 (Section 6.1). Each node also periodically exchanges the MVE parameters with their neighbors. The collected MVE parameters are used by the *Replication Factor Decision* module to estimate replica-

tion gain, and by the *Packet Forwarding* module to measure the delivery capability for a potential next hop carrier.

The *Replication Factor Decision* module collects MVE parameters as well as packet delay feedbacks in order to estimate replication gain, which is then used to calculate the replication factor probability distribution as presented in Section 6.2. The detail of how the Replication Factor Decision works is presented in the following secion. When a new packet arrives, the *Packet Forwarding* module draws a replication factor using the probability distribution produced by this module.

Finally, the *Packet Forwarding* module implements the forwarding strategy. The details are presented in the following sections. Here we highlight that our forwarding strategy utilizes MVE parameters for measuring delivery capability, which incorporate the potential correlation between different nodes. We show its superior performance when compared to other protocols in Section 6.4.

In the following sections, we introduce the details for Replication Factor Decision and Packet Forwarding. We omit MVE Estimation since it has been presented in Section 6.1.

6.3.2 Replication Factor Decision Module

6.3.2.1 Calculation of Replication Factor Probability Distribution

In order to calculate the replication factor probability distribution, we need accurate information of packet delays for any given replication factors to derive corresponding replication gains. One way to achieve this goal is through direct feedback from destination nodes. Although this provides the most accurate delay information, it is infeasible in a DTN environment, due to the fact that: i) packets may not be delivered, and ii) even if packets are delivered, high round trip time prevents nodes from acquiring up-to-date information. Another option is to estimate packet

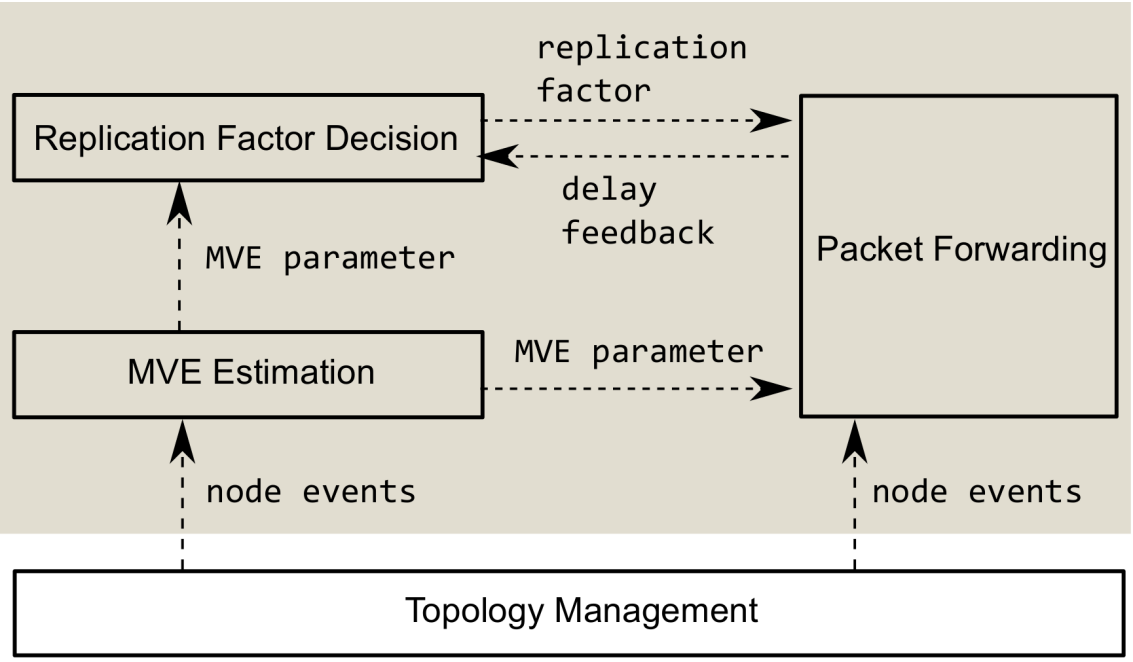


Figure 6.9: HRP architecture

delays through a modeling approach. In our case we can use the collected MVE information, as it also takes into account the potential delay correlations. This approach, although less accurate than direct feedback, still provides useful information for estimating replication gains in a DTN environment.

Given that these two approaches have advantages in different environments, we calculate the replication factor probability distribution $\boldsymbol{\pi}$ using delays acquired from both approaches, and combine them in a weighted summation form as follows:

$$\boldsymbol{\pi} = p_c \cdot \boldsymbol{\pi}_m + (1 - p_c) \cdot \boldsymbol{\pi}_d$$

$\boldsymbol{\pi}_m = \{\pi_{m,r}\}_{r=1}^{r_{max}}$ is the probability distribution calculated through delays feedback collected from destination. A PI PW algorithm is used to track the delay for each packet, and to update each element of $\{\pi_{m,r}\}_{r=1}^{r_{max}}$ accordingly.

$\boldsymbol{\pi}_d = \{\pi_{d,r}\}_{r=1}^{r_{max}}$ is calculated through delay estimation using MVE information. At its core it is a FI PW algorithm as the delay estimation can be obtained for each replication factor at the same time.

$p_c \in [0, 1]$ is the credibility score for using $\boldsymbol{\pi}_m$, which is given by:

$$p_c = \frac{T_c}{T}$$

T_c denotes the total time during which the destination and the source are connected in the same network, and T denote the total time elapsed. Notice that $p_c \to 1$ if two nodes are connected all the time, indicating a mesh network-like environment; while $p_c \to 0$ indicates a DTN environment. If p_c does not converge to either value, then although routing protocols designed for mesh networks might work sometimes, the network is still unstable. Depending on the preference parameter α, our algorithm may still use replication to improve packet delivery.

6.3.2.2 Delay Estimation in DTN Environments

Accurate delay estimation in DTNs for a routing protocol given a replication factor is complex even for scenarios with homogeneous and independent node mobility [79]. In our case, we have to consider both heterogeneous and correlated node mobility. We therefore approximate the delay by its upper bound. To facilitate our analysis, we make two simplifying assumption: a) only the source node may replicate packets; and b) destination is not met until all replicas are distributed.

Consider a source node using replication factor r, which has a set of MVE parameters $\{\lambda_{\boldsymbol{s}}^{src}\}$ and has also collected destination's parameters $\{\lambda_{\boldsymbol{s}}^{dst}\}$. The basic idea is to calculate the expected delay when the source randomly replicates $r-1$ copies during its first $r-1$ contact events, after which the packet gets delivered if one of the r nodes (including the source itself) meets the destination.

Notice that the contact process between a node and all other nodes it has met is a Poisson process with a total rate $\sum_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{src}$ according to the MVE model. Therefore, the total contact rates between the source and other nodes excluding the destination is $\sum_{\boldsymbol{s}:s_{dst}=0} \lambda_{\boldsymbol{s}}$. Assuming the source node replicates one copy during each contact, the delay for replicating $r-1$ copies is $\frac{r-1}{\sum_{\boldsymbol{s}:s_{dst}=0} \lambda_{\boldsymbol{s}}}$.

Since $r-1$ nodes are randomly selected as relays, the total contact rates of these r nodes to the destination is $\sum_{\boldsymbol{s}} p_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{dst}$, where $p_{\boldsymbol{s}}$ is the probability that at least one of the nodes in group $\boldsymbol{s}$ is selected as a relay. If we denote p_{s_i} as the probability that node i is selected as relay, then $p_{\boldsymbol{s}} = 1 - \prod_{i:s_i=1}(1-p_{s_i})$. Notice that $p_{s_i} = 1$ if i is the source. To calculate p_{s_i} for other nodes, first consider a single contact event for the source node. Node i is selected if the source meets one group that i belongs to and selects i as relay. The probability for this event is $\sum_{\boldsymbol{s}:s_i=1} \frac{\lambda_{\boldsymbol{s}}^{src}}{\sum_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{src}} \cdot \frac{1}{|\boldsymbol{s}|}$, where $\frac{\lambda_{\boldsymbol{s}}^{src}}{\sum_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{src}}$ is the probability that group $\boldsymbol{s}$ is met and $\frac{1}{|\boldsymbol{s}|}$ is the probability that node i is selected. Then

$$p_{s_i} = \sum_{c=1}^{r-1} (1 - \sum_{\boldsymbol{s}:s_i=1} \frac{\lambda_{\boldsymbol{s}}^{src}}{\sum_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{src}} \cdot \frac{1}{|\boldsymbol{s}|})^{c-1} \cdot (\sum_{\boldsymbol{s}:s_i=1} \frac{\lambda_{\boldsymbol{s}}^{src}}{\sum_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{src}} \cdot \frac{1}{|\boldsymbol{s}|})$$

Using the above equation we are able to calculate $p_{\boldsymbol{s}}$. Therefore, the delay for r nodes to deliver the packet is $\frac{1}{\sum_{\boldsymbol{s}} p_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{dst}}$.

Combined with the delay when source replicates copies, the total delay for replication r is $\frac{r-1}{\sum_{\boldsymbol{s}:s_{dst}=0} \lambda_{\boldsymbol{s}}} + \frac{1}{\sum_{\boldsymbol{s}} p_{\boldsymbol{s}} \lambda_{\boldsymbol{s}}^{dst}}$.

6.3.3 Packet Forwarding Module

The created packet copies specified by the Replication Factor Decision module need to be forwarded to appropriate packet carriers (the selection of which should account for the correlation among different nodes since it impacts the delivery ca-

pability as shown in Section 6.1). We therefore use the MVE parameters and define a new forwarding metric called *additional contact rate* (ACR) that depends on the existing set of packet carriers. Let $\boldsymbol{c}$ denote the current set of packet carriers, and $C = \{j : c_j = 1\}$.

Definition 12. *ACR for node i to destination dst given packet carrier set $\boldsymbol{c}$:*

$$ACR_{v|\boldsymbol{c}} = \sum_{\boldsymbol{s}:s_i=1,s_j=0,j\in C} \lambda_{\boldsymbol{s}}^{dst}$$

Notice that $ACR_{v|\boldsymbol{c}}$ accounts for the correlation of ICTs to the destination, between v and the existing set of carriers. If v is highly correlated, $ACR_{v|\boldsymbol{c}}$ is likely to be small since the *additional* contribution made by v is little.

To facilitate the forwarding strategy for the HRP, we augment the packet header with four new fields.

- $nrofCopies$ is the remaining number of data copies.
- $next\_carrier$ specifies the intermediate destination for the packet. The current carrier sets this field when it decides to forward the packet to that node.
- $carriers$ contains a list of packet carriers that currently hold this packet. This field facilitates the forwarding strategy to evaluate the ACR metric of a node.
- $path$ contains a list of nodes that specifies a shortest path leading to the $next\_carrier$ in the current network.

We present the forwarding strategy in Algorithm 5. As shown, if the destination is in the same network with v, we quickly send the packet along $nrofCopies$ paths (line 3 to 6). This allows us to take advantage of the replication to deal with the potential lossy mesh network which v is connecting to. Notice that if the network is

Algorithm 5 HRP Forwarding Algorithm For Node v

```
1: Neighbors ← TopologyManagement.Neighbors
2: for all Packet m do
3:     if m.dst is in Neighbors then
4:         m.next_carrier ← m.dst
5:         Find m.nrofCopies shortest paths to m.dst
6:         Send m along each path
7:     else if m.nrofCopies > 1 then
8:         c ← m.carriers, c' ← m.carriers\{v}
9:         u ← arg max_{u∈Neighbors} ACR_{u|c}
10:        if ACR_{u|c} > ACR_{v|c'} then
11:            Duplicate m' ← m
12:            m'.next_carrier ← u
13:            m'.path ← ShortestPath(u)
14:            m'.nrofCopies = m'.nrofCopies − 1
15:            Send m' along m'.path
16:            Update m.nrofCopies = 1
17:        end if
18:    end if
19: end for
```

well connected, the $nrofCopies$ output by the learning algorithm has high probability to be 1, and thus reduces to a single shortest path forwarding. If the destination is not present and $nrofCopies > 1$, we send $nrofCopies - 1$ copies to the best possible carrier according to the ACR metric (line 7 to 18). Note that only nodes with more than 1 copy can continue to replicate the packet. This allow the HRP to keep track of the current packet carriers and evaluate the ACR metric.

When a node receives a packet, it first checks if it is the destination. If it is the destination, the node sends back an ACK containing the delay of the packet. Otherwise, it adds itself to the *carriers* if it is the *next_carrier* or simply forwards the packet to the next hop according the *path* field if it is not the *next_carrier*.

6.4 Simulation Evaluation

In this section, we thoroughly evaluate our proposed protocol through extensive simulations in the ONE simulator [42]. We simulate our protocols in three types of scenarios spanning a wide range of network conditions, i.e., well connected mesh networks, sparsely connected DTNs and networks with gradually changing connectivity characteristics.

6.4.1 Simulation Setup

For simulating well connected mesh networks, we generate Random Geometric Graphs (RGG) with 30 nodes in an $250 \times 250m^2$ area. Each node's radio range is set to $100m$. The links formed among nodes have a packet loss probability p. Three data flows are generated with source and destination selected randomly. Each source generates packets with data size of 50kB and with a flow intensity of 6 packets/minute. The total simulation time is 20,000 seconds.

To simulate DTN environments, we conduct trace-driven simulations using real world mobility trace, i.e., Reality [22]. In this case, sources and destinations are selected randomly from all the nodes. A total number of 5,000 packets are generated and each packet has a data size of 500kB.

In order to further test our protocol in networks with diverse connectivity, we gradually change the connectivity for the mesh networks generated by the RGG model by turning each node on and off randomly according to a Poisson process and varying the proportion of time during which each node is online. We refer to this scenario as on-off network. Notice that this scenario is also used in [80] to stress-test the hybrid protocol under diverse connectivity characteristics. Other setups are identical to the first scenario of a well connected mesh network.

For all simulations, the bandwidth of nodes are set to 2Mbps. The first half of

the simulation is used as a warmup period, and data are collected during the second half of the simulation.

By default, our protocol is implemented with $r_{max} = 8$ and $\alpha = 0.5$. r_{max} is chosen such that the protocol has a wide range of choices for the replication factor. $\alpha = 0.5$ is chosen to balance the performance and overhead. We also test with different r_{max} and α. We compare our protocol with the following protocols:

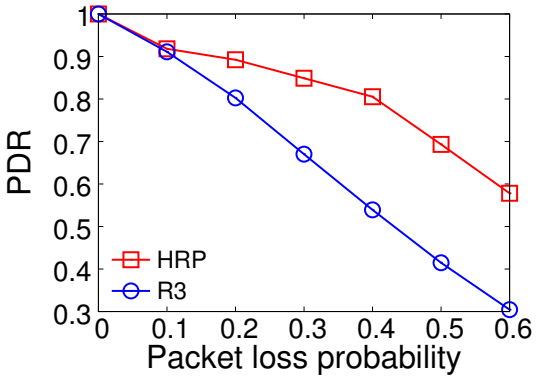
- R3 [80]: is the state-of-art hybrid routing protocol, which uses replication gain as an important metric. We implement the protocol such that two paths are used if the replication gain is larger than 1.1 [80].

- CIT [8]: is the state-of-art DTN routing protocol that utilizes correlated node mobility to measure delivery capability. CIT stands for *conditional inter-contact time*, which is the average inter-contact time between two nodes relative to meeting with a third node.

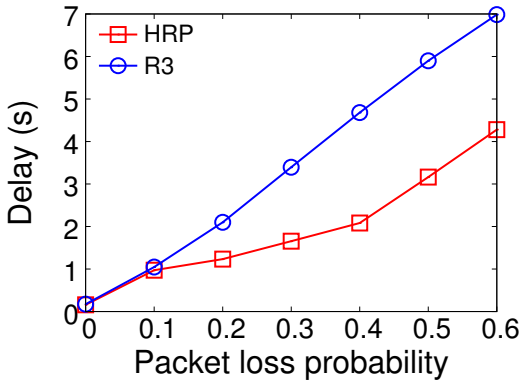
- Spray&Wait [79]: is a DTN routing protocol which limits the packet replications. A fixed number of packet replicas are first "sprayed" into the network. The nodes who receive these packets then "wait" until the destination is met and then the packet is delivered. Here we set the spray counter to 6, and use the binary spray mechanism.

- Epidemic [82]: is essentially a flooding-based protocol. We use it as a baseline comparison.

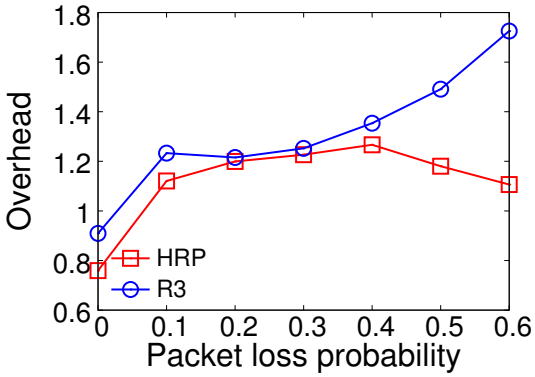
We evaluate all protocols in terms of packet delivery ratio (PDR), delay and overhead. The delays are calculated such that undelivered packets have delays as the packet deadline, which is set to $10s$ in mesh network, 1 hour up to 2 weeks in DTN



(a) PDR in mesh



(b) Delay in mesh



(c) Overhead in mesh

Figure 6.10: Performance evaluation in a mesh network with lossy links

and 5 minutes in the On-off network. The overhead is defined as $\frac{\#relays - \#delivered}{\#delivered}$, which reflects how efficient the routing protocol is in terms of packet delivery.

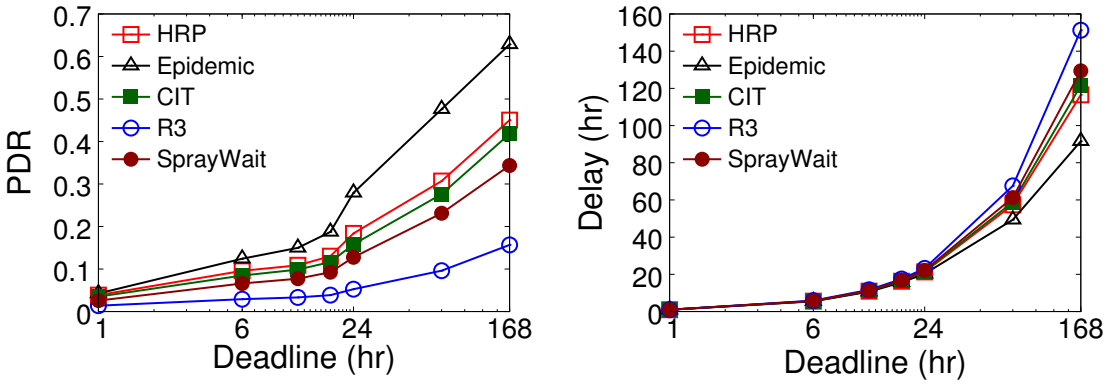
6.4.2 Evaluation in Lossy Mesh Networks

A well connected mesh network becomes disconnected when the packet loss problem becomes severe. An appropriate amount of replication is able to improve the data delivery reliability. We vary the packet loss probability p up to 0.6 to stress-test our protocol and compare it with the R3 protocol. The results are presented in Figures 6.10. As we can see from the figures, our protocol is able to improve both PDR

and delay by up to 1.9x and 2.2x, respectively, when compared to the R3 protocol. The main reason is that R3 progressively discards paths, which may however be candidates of alternative paths, to reduce computational cost and it overestimates the replication gain of using two paths when in fact more paths are beneficial to the delivery performance. On the contrary, HRP is able to choose replication factors from a larger range of values. The direct delay feedback plays an important role in terms of accurate estimation of replication gain, which leads to better decisions of the replication factors. As a result, HRP delivers more packets with less delays without incurring too much overhead.

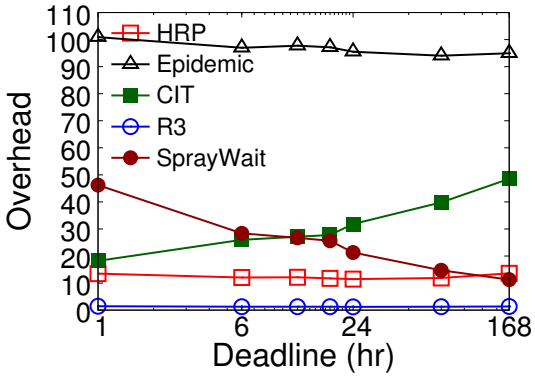
6.4.3 Evaluation in DTN

We use deadline as the main parameter for evaluating our protocol in DTNs as it is used in many state-of-art DTN protocols [92] [8]. We vary the deadline up to 2 weeks and present the results in Figure 6.11. First, we notice that HRP achieves the second best delivery ratio and delay, and improves upon R3 by up to 3.5x and up to 23%, respectively. R3 performs poorly in DTN environments due to its source routing technique. As the entire route is specified in the packet header, intermediate nodes cannot make additional routing decisions and thus cannot take advantage of opportunistic contacts where the encountered nodes may have even better delivery capability. Second, HRP delivers more packets when compared to CIT, which is a state-of-art DTN protocol that utilizes correlated mobility, but with only 1/2 of the overhead. Essentially the CIT protocol only considers correlation between pairs of nodes and does not limit replication. This results in unnecessary replication and poor packet carrier selection, and thus higher overhead. Third, compared to Spray&Wait which has a fixed number of packet replication, our protocol delivers more packets with less overhead. This demonstrates that the selection of packet carriers using the



(a) PDR in DTN

(b) Delay in DTN



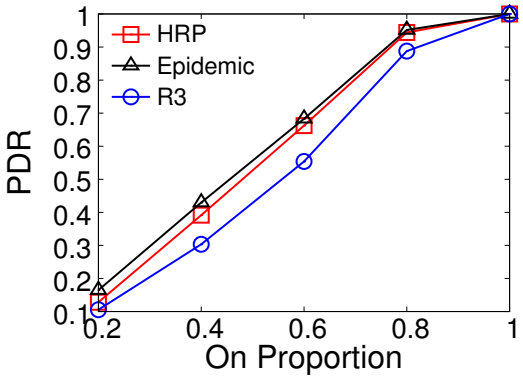
(c) Overhead in DTN

Figure 6.11: Performance evaluation in a DTN environment using the Reality mobility trace

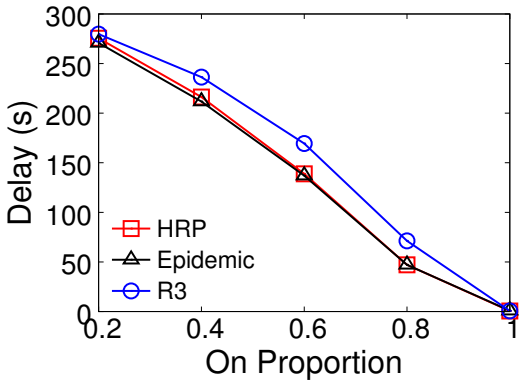
proposed metric is more effective in such a heterogeneous network.

6.4.4 Evaluation in Diverse Network Conditions

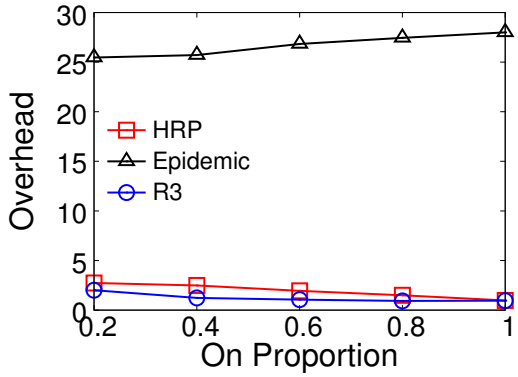
To further stress test our protocol in diverse networks, we turn each node in the mesh network used in Section 6.4.2 on and off according to a Poisson process with mean inter-event time of 10 minutes. During each interval, the node is turned on for a proportion of time and then shut down. Similar experiments were performed in [80]. In this scenario, the packet deadline is set to 5 minutes. The results are



(a) PDR in OnOff Network



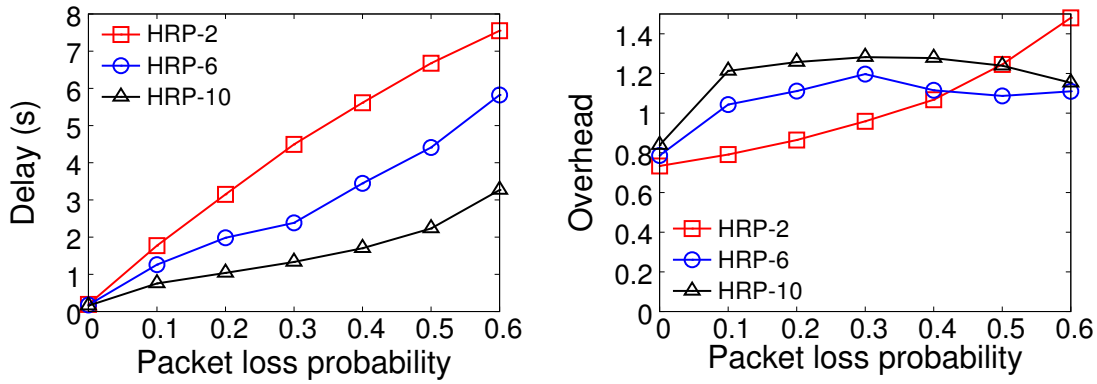
(b) Delay in OnOff Network



(c) Overhead in OnOff Network

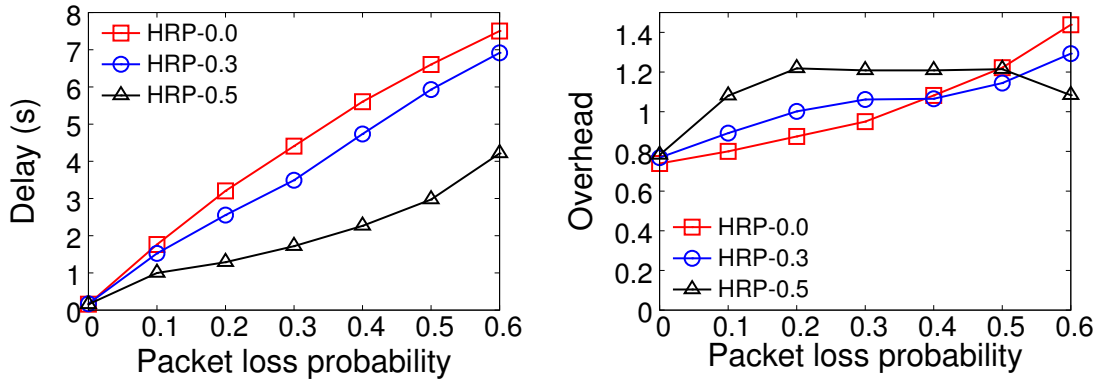
Figure 6.12: Performance evaluation in a dynamic network environment where nodes are turned on and off randomly

presented in Figure 6.12. As can be seen from the figures, HRP achieves similar performance in terms of PDR and delay, when compared to Epidemic. It improves PDR by up to 29% and reduces delay by up to 34% when compared to R3. It also uses comparable overhead as R3, which is significantly smaller than Epidemic. Notice that the Epidemic protocol is optimal in terms of delay as it delivers the most number of packets. This demonstrates that HRP is able to adapt itself and deliver satisfactory performance without incurring too much overhead even in this type of dynamic network environment.



(a) Delay

(b) Overhead



(c) Delay

(d) Overhead

Figure 6.13: Performance evaluation for HRP in lossy mesh networks with different r_{max} ((a), (b)) and α ((c), (d))

6.4.5 Evaluation for Different Parameters

We study the effect of r_{max} and α by adjusting their value and run the same set of simulations. Recall that r_{max} is the range of replication factor that our learning algorithm can select from. α indicates the preference for optimizing delay performance. Due to space limitation, we only present the results for the mesh networks in Figure 6.13. In Figure 6.13a and 6.13b, we vary r_{max} from 2 to 10. Notice that when $r_{max} = 10$ the protocol replicates more aggressively and has much better performance

in terms of delay but with higher overhead. The additional replication starts to pay-off, i.e., overhead drops, when the packet loss probability is high. For $r_{max} = 2$, the replication is limited, but its overhead exceeds the protocol with $r_{max} = 10$ when the packet loss is severe. In Figure 6.13c and 6.13d, we vary the α from 0.0 to 0,5. Notice that when $\alpha = 0.0$, it is essentially a shortest path based single copy protocol. By increasing α, HRP starts to replicate when packet loss probability increases. The overhead of the protocol when $\alpha = 0.5$ is the same and even smaller than when $\alpha = 0.0$ if the packet loss probability is high, and it is able to achieve only half of the delay.

r_{max} and α provides a tool for users to adjust the preference between optimizing performance and reducing resource utilization. The protocol itself can then adapt itself according to the network condition it currently deals with.

6.4.6 Summary of Simulation Results

From the simulation results we can see that HRP is able to adapt itself according to different network conditions. The Replication Factor Decision module reacts to different network conditions by selecting replication factors from a wide range of values to satisfy the performance and resource utilization trade-off. The DTN scenario also validates the effectiveness of selecting packet carriers using MVE estimation based ACR metric. As shown in the figures, HRP achieves better performance in terms of PDR and delay, while it uses smaller overhead than both Spray&Wait and CIT.

6.5 Proof-of-Concept Implementation and Evaluation

We implement a proof-of-concept prototype of HRP and deployed it on Mikrotik RB433UAH routers to evaluate its performance. In this section, we introduce the implementation details and present the real world evaluation results.

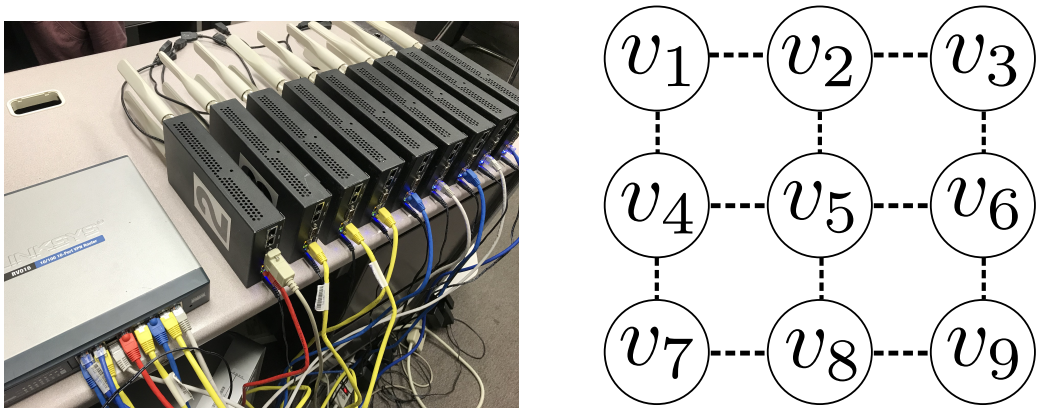
6.5.1 HRP Implementation

We implement HRP in C++ as a user-space daemon process. The daemon handles both control traffic and the actual data forwarding. Notice that we choose to implement at user space due to the ease of development and maintenance. The disadvantage is, however, that each packet has to travel through the border between user space and kernel space, which inevitably increases the processing delay. Although the optimization of coding could further improve the performance, we leave it as future work as it is out of the scope of this section.

We leverage Optimized Link State Routing (OLSR) [38] for topology maintenance. Instead of broadcasting and collecting beacons to maintain topology, the HRP daemon invokes *olsrd* [81], an OLSR implementation, during the initialization phase and fetch topology information (the graph view of the current connected network) from it.

HRP daemon exchanges MVE information once they discover each other. In addition to that, the daemon also probes to each discovered neighbor every t_{probe} seconds in order to collect delay information for different replication factors. The probes are sent along r paths specified by the Replication Factor Decision module. We implement Yen's algorithm [90] for finding multiple shortest paths to the destination. Once the neighbor receives the probe, it immediately sends back a reply. Upon the reception of the reply, the sender then feeds the round-trip time (RTT) to the Replication Factor Decision module to calculate replication factor distribution. In our experiments, we set t_{probe} to 2 seconds.

As the replication factor distribution $\boldsymbol{\pi}_m$ may converge, additional probes are no longer necessary. To save bandwidth, we adopt a *slow reduce, fast recover* approach to adjust the t_{probe}. We set t_{probe} to $2 \cdot t_{probe}$ each time when the absolute difference



(a) Testbed for proof-of-concept implementation and evaluation

(b) Network topology for the fully connected mesh network

Figure 6.14: Experiment setup

between $\pi_{m,r}$ and its previous value is within 1%, for all r. Otherwise, we set the t_{probe} back to the original value, i.e., 2 seconds.

6.5.2 Experiment Setup

We deploy the HRP daemon on 9 Mikrotik RB433UAH routers, which have AR7161 CPU operating at 680MHz and have 128MB RAM. All the routers run OpenWRT 15.05. The wireless cards are set to operate in 5GHz (channel 44) and in ad-hoc mode. Due to space limitation and for the ease of testing, we place all the routers together and emulate a fully connected multi-hop mesh network by manipulating the firewall configurations. To this end, we connect all the routers to a server through Ethernet cable and issue *iptables* commands from the server to create different topology. The testbed setup is shown in Figure 6.14a.

We conducted two sets of experiments, i.e., in a fully connected mesh network and a dynamic on-off network. For the fully connected mesh network, we created a topology as shown in Figure 6.14b. For each experiment, we generate two data flows with inter-message arrival time set to 5 seconds. We vary the data size from 8 kB

up to 32 kB, and set the deadline of each data to 30 seconds. For on-off network, we use the same baseline topology as shown in Figure 6.14b but turn each node on and off randomly with different on-proportion. The expected total duration for one on-off cycle is set to 60 seconds. We generate two data flows with size set to 8 kB and with inter-message arrival time set to 30 seconds. We set the deadline of each data to 300 seconds.

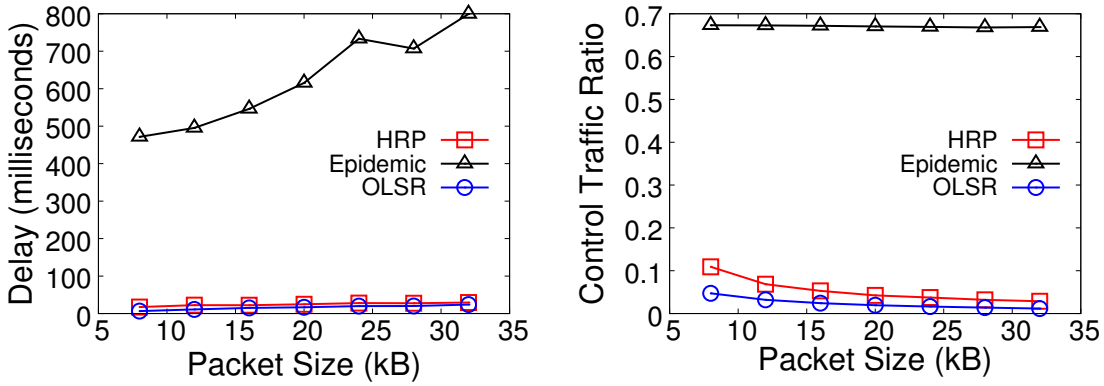
In addition to HRP, we also run OLSR and Epidemic for comparison. Notice that these two protocols are designed with specific network condition in mind. We intends to show that 1) they perform poorly when outside their designed scenario, and 2) that HRP perform closely to the best protocol for each specific scenario. For OLSR we use the *olsrd* implementation, and for Epidemic, we adopt the IBR-DTN [21] implementation.

Finally, in order to understand how HRP dynamically choose the replication factor for each data, we run additional experiments which logs how the replication factor distribution outputted by the Replication Factor Decision module evolve with time. Notice that logging is by default turned off, as it otherwise negatively impacts the performance.

We are interested in the same set of metrics as in the simulation study, i.e., PDR, PDD and overhead. In addition to these, we also collects the information of *control traffic ratio*, defined as $1 - \frac{\text{size of payload transferred}}{\text{size of total traffic}}$ for the fully connected mesh network.

6.5.3 Evaluation in Fully Connected Mesh Network

Figure 6.15 shows the results in fully connected mesh network. We omit the results for PDR as it is trivially one for all protocols. As we can see from Figure 6.15a, HRP achieves comparable delays with OLSR. Specifically, the delays for OLSR ranges from 6.5 ms up to 23.99 ms, while the delays for HRP ranges from 17.46 ms up to



(a) Delay

(b) Control Traffic Ratio

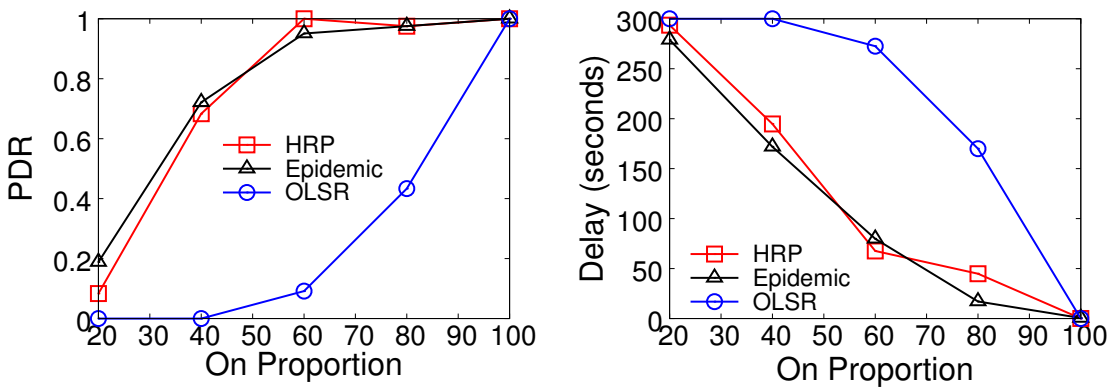
Figure 6.15: Performance evaluation for HRP in well connected mesh networks with different packet size

29.48 ms. On the other hand, we can see that Epidemic has delays ranges from 471.49 ms up to 799.36 ms, an order of magnitude larger than HRP and OLSR.

As for control traffic overhead, we can see that HRP is slightly higher than OLSR, while also significantly smaller than Epidemic. HRP has control traffic ratio ranges from 10% down to 2.8%, whereas OLSR ranges from 4.7% down to 1.1%. Notice that the increased overhead is the price HRP has to pay in order to adapt to the network condition. As we will see in the following section, HRP's ability to adapt to network condition leads to superior performance in dynamic networks.

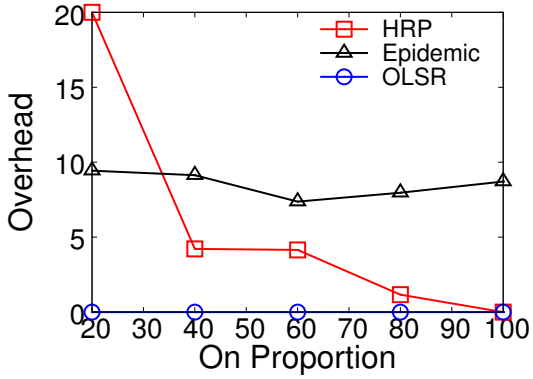
6.5.4 Evaluation in Diverse Network Conditions

Figure 6.16 shows the results in the on-off network. Similar to the results in simulations, HRP performs very close to Epidemic in terms of PDR and PDD under different on-proportions. When the on-proportion is 20%, Epidemic achieves PDR 18.8%, whereas HRP's PDR is 8.3%. Hence in extremely sparse connectivity networks, Epidemic's flooding strategy is still efficient. However, when on-proportion ranges from 40% to 100%, HRP's PDR results are within 3% difference when com-



(a) PDR

(b) Delay

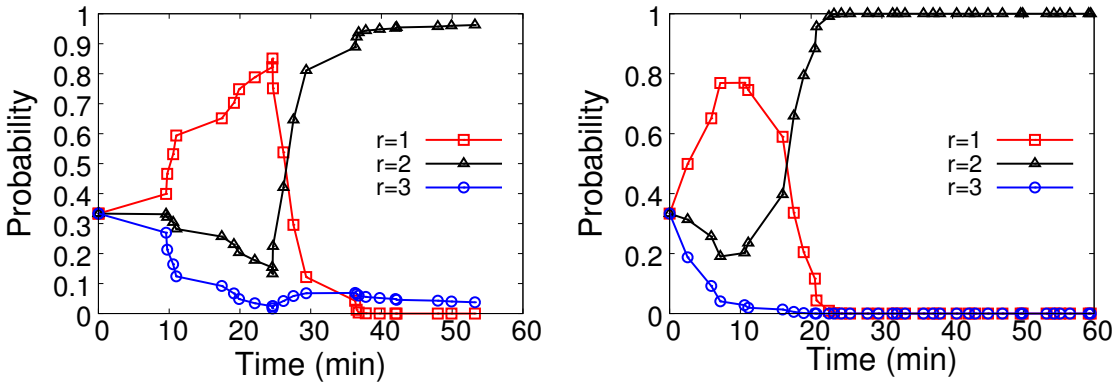


(c) Overhead

Figure 6.16: Performance evaluation for HRP in a on-off network with different on-proportion

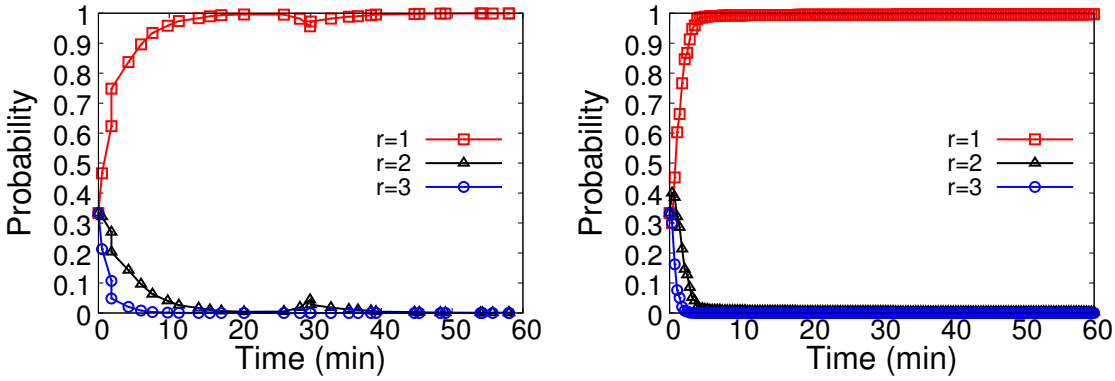
pared to Epidemic. Similar results can be observed for delays as well. Notably, HRP achieves these results but with only half or less overhead compared to Epidemic.

On the other hand, it is clearly that OLSR is not designed for the environments when the on-proportion is not 100%. As we can see from the figures, OLSR achieves significantly lower PDR and significantly higher PDD when compared to HRP and Epidemic. Although OLSR has zero overhead as it does not generate redundant traffic under all scenarios, it does not efficiently deliver packets either.



(a) Distribution of $\boldsymbol{\pi}_d$ when on-proportion is 40%. The p_c value is 0.033 in this scenario

(b) Distribution of $\boldsymbol{\pi}_d$ when on-proportion is 60%. The p_c value is 0.199 in this scenario



(c) Distribution of $\boldsymbol{\pi}_d$ when on-proportion is 80%. The p_c value is 0.807 in this scenario

(d) Distribution of $\boldsymbol{\pi}_m$ when mesh network is fully connected. The p_c value is 1.0 in this scenario

Figure 6.17: Convergence of replication factor distribution under different network conditions

6.5.5 Convergence of Replication Factor Distribution

Recall that Replication Factor Decision module calculates the replication factor distribution $\boldsymbol{\pi}$ based on two distributions $\boldsymbol{\pi}_m$ and $\boldsymbol{\pi}_d$ and one p_c value, where $\boldsymbol{\pi}_m$ is based on delay feedbacks, $\boldsymbol{\pi}_d$ is based on the modeling of delay, and p_c is the proportion of time when the node is connected to the destination. In Figure 6.17 we demonstrate how these distributions evolve with time under different scenarios.

Figures 6.17a to 6.17c show the results of $\boldsymbol{\pi}_d$ when the on-proportion ranges from 40% to 80%. As we can see from the figures, when the connectivity is more sparse, i.e., on-proportion is 40% and 60% and p_c ranges from 0.033 to 0.199, ultimately $P(r = 2)$ converges to 1.0. That is to say, the source will mostly use replication factor $r = 2$ for all the packets created. Notice that this does not necessarily mean that the source will always generate two copies for a packet, as the forwarding is also depended on the connectivity and the forwarding strategy. When the on-proportion is 80%, p_c is 0.807, and $P(r = 1)$ converges to 1.0 fairly quickly. This is also intuitive, since in this case it may not be beneficial to replicate.

Figure 6.17d shows the results of $\boldsymbol{\pi}_m$ when the network is fully connected. In this case, the $\boldsymbol{\pi}$ is completely decided by $\boldsymbol{\pi}_m$ as $p_c = 1.0$. As we can see, $P(r = 1)$ converges to 1.0 within just a few minutes. This indicates that, after the initial phase, HRP performs very similar to OLSR which is optimized for the connected network.

7. COST-AWARE ENERGY EFFICIENT MOBILE DATA OFFLOADING

The rapid advance of hardware and the explosion of smartphone applications are driving the surging demand of data access for end users. However, despite the ubiquitous network coverage, communication over cellular network incurs non-negligible monetary cost. Additionally, cellular communication is less energy efficient when compared to WiFi due to various reasons such as high tail energy [32].

As many data transfers are delay tolerant (e.g., video uploading [68]), *delayed data offloading* to WiFi Access Points (AP) [5] [47] [56] has been seen as a promising solution to help reduce energy consumption and monetary cost. Delayed data offloading refers to the technique where users intentionally postpone the delay-tolerant data transfer in the hope of meeting some WiFi AP. If no AP is met within the given deadline, the cellular network is used to complete the transmission. It may help users reduce monetary cost, as cellular usage might be avoided. Additionally, it may help users reduce energy consumption, since the user may have a more energy efficient transmission to a WiFi AP [68]. The tradeoff, however, is the increased delay.

In this section, we focus on the *cost-aware energy efficient offloading problem* in which users upload data to a remote server, within a given deadline and with a monetary cost constraint, while trying to minimize the energy consumption. This problem applies to many real world application scenarios [68] [53]. Although there are existing solutions trying to solve similar problems [5] [68] [36] [26] [53], a satisfactory one is still missing. Some works [68] [36] ignore *opportunistic forwarding* among mobile devices. However, existing research has shown that the proliferation of mobile devices create frequent contacts that can be leveraged for data forwarding [80] [89]. Ignoring high contact rates among mobile nodes may lead to lower offloading success

rate, and hence higher energy consumption and monetary cost. Moreover, limiting offloading to AP may result in energy consumption and monetary cost, which are coupled when we adjust deadline (i.e., shorter deadline indicates lower offloading success rate and hence simultaneously higher energy consumption and monetary cost [56]). Other research [53] [51] consider opportunistic forwarding among mobile devices. However, they do not explicitly consider energy consumption and monetary cost which can be important to the users.

Designing a coherent offloading strategy that simultaneously considers opportunistic forwarding, energy consumption and monetary cost is challenging. In particular, opportunistic forwarding may incur more energy consumption when compared to direct WiFi offloading since more transmissions are involved. On the other hand, such forwarding may lead to higher offloading success rate which may, in turn, reduce both energy consumption and monetary cost. Consequently, in order to take advantage of the opportunistic forwarding while avoiding unnecessary overhead, we ask the following question: *when should a node forward the data, to which node, and when should it stop forwarding?*

To answer the above question, we formulate the cost-aware energy efficient offloading problem as a *discrete time optimal control problem*. To solve the problem, we show that the optimal policy actually has a simple threshold-based structure: when the node encounters another node, if it observes that the energy consumption for transferring data is smaller than a certain threshold, it forwards the data. We present a dynamic programming based algorithm to construct the optimal policy. Unfortunately, due to the curse of dimensionality, the algorithm runs in exponential time, which is not practical for solving real world problems. To this end, we propose a novel *one step lookahead* based policy, named Cost-aware Energy efficient Offloading (CEO) policy, to approximate the optimal policy. CEO can be constructed efficiently

in pseudo-linear time. Through extensive trace-based simulation, we show that CEO adapts well to the environments and outperforms state of art protocols. If the user would like to reduce monetary cost, CEO achieves up to 64% higher offloading success rate, with reasonable increase or comparable energy consumption. If the user can afford cellular monetary cost, CEO achieves comparable, if not better, offloading success rate, but with up to 23% energy consumption reduction.

This section is organized as follows. In Section 7.1 we introduce the system model. We mathematically formulate the problem in Section 7.2. Then we present the design of the one step lookahead based approximation policy in Section 7.3. Simulation setup and performance evaluation results are presented in Section 7.4.

7.1 System Model and Problem Statement

In this section, we first present the system model. Then we present the problem through a simple example. A formal mathematical formulation of the problem is presented in Section 7.2.

7.1.1 System Model

We consider a system where the source node S has a *message* to be transmitted to a remote server D within a given deadline T. Time is slotted with a time slot duration δ_t, hence the deadline is also referred to as time horizon $N = \lfloor \frac{T}{\delta_t} \rfloor$. S may offload the data via opportunistic communication, or if the deadline is reached it has to transmit using the cellular network. Node S meets with Access Points (AP) and a set of n mobile nodes $\{v_i\}_{i=1}^n$ opportunistically. Meanwhile, nodes $\{v_i\}_{i=1}^n$ also meet with APs. The APs are assumed to have Internet connection and can connect to D. We therefore abstract APs as one super node in the system. The contact processes are assumed to be Poisson processes, with contact rates between S and APs, S and v_i, and v_i and APs as λ_0, $\{\lambda_i^m\}_{i=1}^n$, and $\{\lambda_i^d\}_{i=1}^n$, respectively. Notice that the

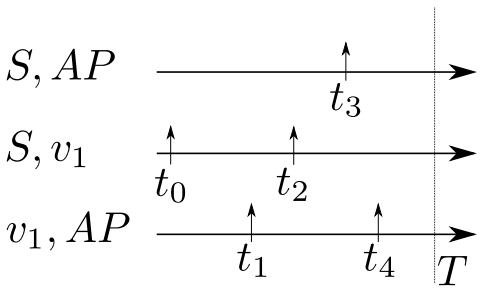
assumption of Poisson contact process between pairs of nodes not only allows for mathematical tractability, but it is also backed by state of the art research which characterizes mobility in opportunistic networks [93] [62] [89].

We focus on the *two-hop* based forwarding strategy in this section, i.e., the mobile nodes which received a copy of the message only forward to APs but not other mobile nodes. We leave the more general forwarding strategy as future work. Due to this reason, and to avoid repetitive definition, we will use subscript 0 to denote variables between S and APs, superscript m to denote variables between S and v_i, and superscript d to denote variables between v_i and APs.

Due to channel condition, data rate, link failures, etc., the energy consumption in terms of *energy per bit* for transmission during each of the opportunistic contacts maybe different [68] [32] [20]. To capture this, we assume that the *energy per bit* during each opportunistic contact are independent random variables. Let e_0, $\{e_i^m\}_{i=1}^n$ and $\{e_i^d\}_{i=1}^n$ denote these random variables, each of which follows the distribution given by the Cumulative Distribution Function (CDF) of F_0, $\{F_i^m\}_{i=1}^n$, and $\{F_i^d\}_{i=1}^n$, respectively. We assume that the *energy per bit* for cellular communication is e_c which follows CDF of F_c.

In addition to the energy consumption, each transmission may incur monetary cost. Notice that there are many different *pricing models* [30] [78]. In this section we consider a simple *usage-based* model, i.e., *cost per bit*. Let us denote the cost per bit for transmissions as c_0, $\{c_i^m\}_{i=1}^n$ and $\{c_i^d\}_{i=1}^n$. Also denote the cost per bit for cellular communication as c_c. For simplicity, in this section we assume that the monetary costs are deterministic. Furthermore, the user may set a cost constraint for transferring a message, which can be converted to a *cost per bit* constraint denoted as c_{cap}.

During each opportunistic contact, we assume that a node can estimate the energy



	e_0	e_1^m	e_1^d
t_0	—	2	—
t_1	—	—	2
t_2	—	3	—
t_3	3.5	—	—
t_4	—	—	1

Figure 7.1: A sample problem where S has to choose whether to transmit during each transfer opportunity in order to minimize the energy consumption, assuming $e_c = 10$.

consumption for transmitting a given data item using information such as Received Signal Strength Index (RSSI), data rates, etc. [68]. Meanwhile, the transmitting node can gather the cost information from the encountered node. We also assume that the node will record this information as history, which can be used in the future for making forwarding decisions.

7.1.2 Problem Statement

Given the time horizon N, contact rates (λ_0, $\{\lambda_i^m\}_{i=1}^n$, and $\{\lambda_i^d\}_{i=1}^n$), energy consumption distributions (F_0, $\{F_i^m\}_{i=1}^n$, $\{F_i^d\}_{i=1}^n$, and F_c), monetary costs (c_0, $\{c_i^m\}_{i=1}^n$, $\{c_i^d\}_{i=1}^n$, and c_c), we need to calculate the optimal *policy*, which maps an observed state (defined in the following section) to a forwarding decision, such that the expected total energy consumption (which includes transmission energy consumption for all opportunistic forwarding and cellular communication - if offloading fails), is minimized while the total monetary cost is smaller than a user defined constraint c_{cap}.

To illustrate the problem, let us consider an example shown in Figure 7.1. The left side shows the contact processes between S, v_1 and APs, while the right side

shows the energy consumption estimation for each transfer opportunity. We omit the costs for each transmission for simplicity. In this example, the *energy optimal* forwarding strategy is for S to forward to v_1 at t_0, and have v_1 forward to the AP at t_4.

The challenge for this problem is two fold. First, a node does not have the full knowledge about the future. When S meets v_1 at t_0, it has no idea when is the next encounter, or with which node, or the energy consumption and monetary cost for future contacts; yet S has to make a forwarding decision. Second, the *policy* has to map every possible state into an optimal action such that the overall expected energy consumption is minimized. As we will show in the following section, the construction of such a policy is nontrivial.

7.2 Optimal Cost-aware Energy Efficient Offloading

In this section, we formulate the *cost-aware energy efficient offloading* problem as a *discrete time optimal control problem*. We show that the optimal solution has a simple threshold-based structure, i.e., the node should only forward if the energy consumption for transferring the message is smaller than a certain threshold.

7.2.1 System Dynamics

We begin with introducing the system dynamics. We use x_k, y_k, z_k to represent the current state of the system at time slot k. The state is observed by the system and is used to make a control decision.

Let $\hat{x_k} = \{x_{k,i}\}_{i=1}^{n} \in \hat{\mathcal{X}}$ represent the state of *message carriers* , where $\hat{\mathcal{X}} = \{0,1\}^n$. We refer to a mobile node as a *message carrier* if it carries a copy of the message. Let $x_k \in \mathcal{X}$, where $\mathcal{X} = \{0,1\}^{n+1}$, be the *augmented* state defined as

follows:

$$x_{k,i} = 1 \iff \begin{cases} i \in \{1, ..., n\}, & v_i \text{ is a message carrier} \\ i = n+1, & \text{the message is delivered} \end{cases}$$

where $\{x_k : x_{k,n+1} = 1\}$ are *terminal* states. Initially, $x_0 = \mathbf{0}$, where $\mathbf{0}$ is a $n+1$ dimensional vector.

Let $y_k = \{y_{k,i}\}_{i=0}^{2n} \in \mathcal{Y}$ denote the current contact event at time slot k, where $\mathcal{Y} \subseteq \{0,1\}^{2n+1}$. y_k is composed of three parts, i.e., $y_{k,0}$, $\{y_{k,i}^m\}_{i=1}^n$, and $\{y_{k,i}^d\}_{i=1}^n$, where the latter two are n-dimensional vectors. y_k is defined as follows:

$$y_k = \begin{bmatrix} y_{k,0} \\ y_k^m \\ y_k^d \end{bmatrix}, \begin{cases} y_{k,0} = 1, \text{ contact between } S \text{ and AP} \\ y_{k,i}^m = 1, \text{ contact between } S \text{ and } v_i \\ y_{k,i}^d = 1, \text{ contact between } v_i \text{ and AP} \end{cases}$$

Let $z_k = \{z_{k,i}\}_{i=0}^{2n} \in \mathbb{R}_+^{2n+1}$ denote the observed energy consumption at time slot k. z_k is also composed of three parts in a similar fashion as y_k, i.e., $z_{k,0}$, z_k^m, z_k^d, where the latter two are n-dimensional vectors.

At each time slot k, the system observes the states x_k, y_k and z_k, and needs to make a control decision, that is, whether or not to forward a copy of the message and to which node. Let $u_k = \mu_k(x_k, y_k, z_k)$, where u_k is the control vector and μ_k is a function that maps the current system state to a control vector. u_k is constrained to take values in the set $U_k(x_k, y_k) \subseteq \{0,1\}^{2n+1}$ that depends on the current system

state. u_k is composed of three parts, i.e., $u_{k,0}, u_k^m$ and u_k^d, defined as follows:

$$u_k = \begin{bmatrix} u_{k,0} \\ u_k^m \\ u_k^d \end{bmatrix}, \begin{cases} u_{k,0} = 1, \text{ forward from } S \text{ to AP} \\ u_{k,i}^m = 1, \text{ forward from } S \text{ to } v_i \\ u_{k,i}^d = 1, \text{ forward from } v_i \text{ to AP} \end{cases}$$

The constrained set U_k is defined as:

$$U_k(x_k, y_k) = \begin{cases} u_k \leq y_k \\ u_{k,j}^d \leq x_{k,j}, j = 1, ..., n \\ u_{k,j}^m \leq 1 - x_{k,j}, j = 1, ..., n \\ c^T \cdot u_k < c_{cap} - \sum_{j=1}^{n} x_{k,j} c_j^m \end{cases}$$

where the first constraint ensures that forwarding only happens when there is a contact; second constraint ensures that the mobile node has to have a copy in order to forward to an AP; the third constraint ensures that S will not forward to the same mobile node multiple times; and the fourth constraint ensures that the forwarding will not exceed the cost constraint, assuming c^T is the cost vector $[c_0, c^m, c^d]$.

With the above notation, we present the system dynamics as follows:

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} h_k(x_k, u_k) \\ w_k \\ e_k \end{bmatrix} \tag{7.1}$$

where h_k represents how the state of message carriers change, and is defined as the

following:

$$h_k(x_k, u_k) = x_k + \begin{bmatrix} u^m_{k,1} \\ ... \\ u^m_{k,n} \\ \max\{u_{k,0}, u^d_{k,1}, ..., u^d_{k,n}\} \end{bmatrix}$$

That is, the first n elements of x_{k+1} is set to 1 if a forwarding from S to a mobile node happens; while the last element of x_{k+1} is set to 1 if the forwarding to an AP happens.

$w_k \in \mathcal{W}_k(x_k) \subseteq \{0,1\}^{2n+1}$ is the random vector which represents the contact events *that can be utilized for forwarding*, i.e., contacts between S and AP, between S and a non-carrier node, and between a carrier node and AP. w_k is composed of three parts, i.e., $w_{k,0}, w^m_k$ and w^d_k. $\mathcal{W}_k(x_k)$ is the set from which w_k takes value. Notice that y_{k+1} is a realization of w_k, which is observed at time slot $k+1$. The probability distribution of w_k depends on the current state x_k, and is calculated as follows. First, the total contact rates for events that can be utilized for forwarding is $\lambda_{total} = \lambda_0 + \sum_{i=1}^{n}(1 - x_{k,i})\lambda^m_i + \sum_{i=1}^{n} x_{k,i}\lambda^d_i$, where the first term is the contact rate between S and AP, the second term is the total contact rate between S and any non-carrier node, and the third term is the total contact rate between carriers and AP. Hence the probability of no event happening is $P(w_k = \mathbf{0}) = e^{-\lambda_{total}\cdot\delta_t}$. Second, we assume that the time slot duration δ_t is sufficiently small such that at one time slot, there is at most one contact event. Consequently, w_k may have at most one nonzero element. The event that one particular contact happens is equivalent to the event that there is one contact and that this particular contact happens first. With slightly abused notation, where we use the nonzero element to represent w_k, we have

the following distribution:

$$P(w_{k,0}=1)=\frac{\lambda_0}{\lambda_{total}}\cdot(1-P(w_k=\mathbf{0}))$$
$$P(w^m_{k,i}=1)=\frac{(1-x_{k,i})\cdot\lambda^m_i}{\lambda_{total}}\cdot(1-P(w_k=\mathbf{0}))$$
$$P(w^d_{k,i}=1)=\frac{x_{k,i}\cdot\lambda^d_i}{\lambda_{total}}\cdot(1-P(w_k=\mathbf{0}))$$

where $i=1,\ldots,n$.

$e_k\in\mathbb{R}^{2n+1}_+$ is the random vector, which represents the energy consumption, and is composed of three parts, i.e., e_0, e^m, and e^d. z_{k+1} is a realization of e_k and is served as an observation to the system.

7.2.2 Objective

At each time slot k, if the control decision is to forward the message, then it incurs energy consumption. Hence we define the cost function at each time slot $k=0,...,N-1$ as follows:

$$g_k(x_k,z_k,u_k)=\begin{cases}0, & \text{if } x_{k,n+1}=1\\ z_k^T\cdot u_k, & \text{otherwise}\end{cases}$$

i.e., if the message is already delivered at time slot k, there is no cost; otherwise, the cost is the energy consumption if there is a forwarding in this time slot.

We also define the terminal cost, i.e., the cost when the system reaches the time horizon N:

$$g_N(x_N)=\begin{cases}0, & \text{if } x_{N,n+1}=1\\ e_c, & \text{if } c_c+\sum_{i=1}^n c^m_i x_{N,i}<c_{cap}\\ Penalty, & \text{otherwise}\end{cases}$$

i.e., there is no cost if the message is delivered; or the cost is the energy consumption of cellular communication if the current monetary cost plus the cellular communication's monetary cost still satisfies the constraint c_{cap}; otherwise, the cost is a large *Penalty* value.

With the cost function defined as above, we have the total cost for the system as $g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, z_k, u_k)$, where x_k, y_k and z_k equals $\mathbf{0}$ (with appropriate dimension) at $k = 0$, and evolves according to Eq. 7.1 for $k = 1, ..., N$.

We consider the class of *policies* that consist of a sequence of functions $\pi = \{\mu_0, ..., \mu_{N-1}\}$, where μ_k maps the state (x_k, y_k, z_k) into control decisions $u_k = \mu_k(x_k, y_k, z_k)$, such that $\mu_k(x_k, y_k, z_k) \in U_k(x_k, y_k)$ for all $k = 0, ..., N-1$. The *cost-aware energy efficient offloading* problem is to find the optimal policy π^* which minimizes the expected total energy consumption shown as follows:

$$\min_{\pi} J_\pi(x_0) = E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, z_k, \mu_k(x_k, y_k, z_k)) \right\}$$

where the expectation is taken over x_k, y_k, z_k, w_k and e_k.

7.2.3 Optimal Solution

As the problem follows the *optimal control problem* framework, the *dynamic programming* (DP) algorithm can be used to solve it optimally [7]. The DP algorithm essentially minimizes the *cost-to-go* functions iteratively in the time-reverse order, i.e., solves the following series of minimization problems from $k = N-1$ to $k = 0$:

$$J_N(x_N) = g_N(x_N), \tag{7.2}$$

$$J_k(x_k, y_k, z_k) = \min_{u_k \in U_k(x_k, y_k)} \underset{w_k, e_k}{E} \Big\{ g_k(x_k, z_k, u_k) + J_{k+1}(f_k(x_k, u_k, w_k, e_k)) \Big\}, k = 0, ..., N-1 \tag{7.3}$$

where f_k is the system equation defined in Eq. 7.1, and J_k is the optimal expected cost-to-go from time k to N. Then if $u_k^* = \mu_k^*(x_k, y_k, z_k)$ minimizes the right side of Eq. 7.3 for every (x_k, y_k, z_k) and for every k, we have the optimal policy $\pi^* = \{\mu_0^*, ..., \mu_{N-1}^*\}$

7.2.3.1 Energy Consumption Threshold

Here we show that the optimal policy to the *cost-aware energy efficient offloading* problem is actually threshold based.

The key observation is that at any time slot k, $U_k(x_k, y_k)$ contains at most two elements, since time slot duration δ_t is sufficiently small such that at most one contact event may happen at a time. Therefore, $U_k(x_k, y_k)$ has $\mathbf{0}$, and at most one more element u'_k. Without loss of generality, we assume the j-th element of u'_k is one if the j-th element of y_k is one. It is trivial to see that if $U_k = \{\mathbf{0}\}$, then $u_k^* = \mathbf{0}$.

Now let us consider the other case, i.e., $U_k(x_k, y_k) = \{\mathbf{0}, u'_k\}$. Let $x'_k = h_k(x_k, u'_k)$, and notice that $x_k = h_k(x_k, \mathbf{0})$. If $y_{k,j} = 1$, then we have $u'_{k,j} = 1$. Notice that $g_k(x_k, z_k, \mathbf{0}) = 0$, and $g_k(x_k, z_k, u'_k) = z_{k,j}$. We have that $u_k = u'_k$ if and only if the following holds:

$$z_{k,j} + \underset{w_k, e_k}{E} \Big\{ J_{k+1}(x'_k, w_k, e_k) \Big\} < \underset{w_k, e_k}{E} \Big\{ J_{k+1}(x_k, w_k, e_k) \Big\}$$

Hence we can say that S forwards only if the energy consumption $z_{k,j}$ is below the

threshold $\theta_k^*(x_k, y_k)$ defined as:

$$\begin{aligned}&\theta_k^*(x_k, y_k) = \\ &\underset{w_k,e_k}{E}\left\{J_{k+1}(x_k, w_k, e_k)\right\} - \underset{w_k,e_k}{E}\left\{J_{k+1}(x'_k, w_k, e_k)\right\}\end{aligned} \tag{7.4}$$

Essentially, we are comparing the expected overall cost between forwarding and not forwarding, and the threshold represents the *energy consumption reduction* when comparing forwarding to not forwarding. Apparently, if the energy consumed for forwarding is smaller than the achievable energy consumption reduction, the overall energy consumption will be reduced, and hence it is better to forward the message.

Interestingly, such a policy allows for adaptive behaviors. In particular, if S does not meet with AP frequently enough, forward the message might significantly reduce the overall energy consumption. Hence the threshold might be high enough to encourage forwarding. On the other hand, if there are already many message carriers, adding another might not reduce the overall energy consumption much. And hence the threshold might be low enough to prevent further forwarding.

With the threshold defined in Eq. 7.4, Eq. 7.3 can be written as:

$$\begin{aligned}J_k(\cdot) =& \mathbb{1}_{z_{k,j}<\theta_k^*(\cdot)}\left\{z_{k,j} + \underset{w_k,e_k}{E}\left\{J_{k+1}(x'_k, w_k, e_k)\right\}\right\} \\ &+ \mathbb{1}_{z_{k,j}\geq\theta_k^*(\cdot)}\underset{w_k,e_k}{E}\left\{J_{k+1}(x_k, w_k, e_k)\right\}\end{aligned} \tag{7.5}$$

7.2.3.2 Policy Construction Algorithm

Finally we present the optimal policy construction algorithm in Algorithm 6. We first initialize variables (line 1), where *thresh* is the final output of the calculated threshold. Then we first set the *costToGo* functions for the terminal state (lines 2 and 4). From line 5, we traverse the time reversely. Notice that *costToGo* always

Algorithm 6 Optimal Policy Construction Algorithm

```
1: Initialize thresh, dp, and costToGo
2: for all (x_N, y_N) ∈ 𝒳 × 𝒴 do
3:     costToGo(x_N, y_N, z) = J_N(x_N) (See Eq. 7.2)
4: end for
5: for k = N − 1, ..., 0 do
6:     for all x_k ∈ 𝒳 do
7:         dp(x_k) ← E_{w_k,e_k}{costToGo(x_k, w_k, e_k)}
8:     end for
9:     for all (x_k, y_k) ∈ 𝒳 × 𝒴 do
10:        if U_k(x_k, y_k) ≠ {0} then
11:            if y_{k,j} = 1 for some j then
12:                x'_k ← h_k(x_k, u'_k)
13:                thresh(x_k, y_k) ← dp(x_k) − dp(x'_k)
14:            end if
15:        else
16:            thresh(x_k, y_k) = 0
17:        end if
18:        costToGo(x_k, y_k, z) ← J_k(·) (See Eq. 7.5)
19:    end for
20: end for
```

saves the cost-to-go function for time $k+1$. At each time slot k, we first calculate the expected energy consumption for each x_k (lines 6 to 8). For each (x_k, y_k) pair (line 9), we calculate the threshold based on Eq. 7.4 (lines 10 to 17). Finally, the *costToGo* function for this time slot is updated (line 18).

The time complexity for Algorithm 6 is $O(N|\mathcal{X}||\mathcal{Y}|)$, where $|\cdot|$ is the cardinality of the set. Unfortunately, for this problem we have $|\mathcal{X}| = 2^{n+1}$ and $|\mathcal{Y}| = 2n+1$, and hence Algorithm 6's time complexity becomes $O(Nn2^n)$. Notice that the time complexity cannot be further reduced, as there are $N(2n+1)2^{n+1}$ *state-time* pairs, and each needs a threshold. As the algorithm runs in exponential time, we cannot use it in realistic scenarios where n could be a few dozens or even hundreds, which calls for the design of an efficient algorithm.

7.3 Approximation Based Cost-aware Energy Efficient Offloading

In the previous section, we present the problem formulation and the optimal policy. Due to curse of dimensionality, the time complexity for obtaining the optimal policy is exponential. In this section, we present the design of an approximation based policy, named Cost-aware Energy efficient Offloading (CEO), which can be constructed in pseudo-linear time.

7.3.1 Basic Idea

The basic idea is to construct a *one-step lookahead* policy using an *approximated* cost-to-go function. Specifically, at each time slot k and for an observed state (x_k, y_k, z_k), we attain the decision by solving the minimization problem:

$$\min_{u_k \in U_k} E\Big\{ g_k(x_k, z_k, u_k) + \hat{J}_{k+1}(f_k(x_k, u_k, w_k, e_k)) \Big\} \tag{7.6}$$

where $\hat{J}_{k+1}$ is an *approximated* cost-to-go function. Hence, we look one step ahead into the future, and use the approximated cost-to-go function to make the decision.

To build the approximated cost-to-go function, we convert the original problem into a *homogeneous* problem, which can be optimally solved in $O(Nn)$ time. Then we convert the state of the original problem into the corresponding state in the *homogeneous* problem, and use the converted state and the cost-to-go function for the *homogeneous* problem to approximate the true cost-to-go function for the original problem. The detail of the conversion and the intuition behind the conversion is given in Section 7.3.2.

In this way, we avoid the need to traverse all the states in order to construct the policy, but only need to solve a simplified homogeneous problem to build the approximated cost-to-go function. The detail of the policy is given in 7.3.3.

7.3.2 Conversion to a Homogeneous Problem

A *homogeneous* problem refers to the one where all the mobile nodes $\{v_i\}_{i=1}^n$ are identical, i.e., $\lambda_i^m = \lambda_j^m, \lambda_i^d = \lambda_j^d, c_i^m = c_j^m, c_i^d = c_j^d$ for all i, j, $\{e_i^m\}_{i=1}^n$ and $\{e_i^d\}_{i=1}^n$ are i.i.d random variables, respectively. In this section, we omit the subscriptions and only use $\lambda^m, \lambda^d, c^m, c^d, e^m$ and e^d to represent the corresponding variable.

Different from the original problem, the identity of the message carrier in a homogeneous problem is no longer important. Hence we can use a single integer $x_k \in \mathcal{X} = \{0, ..., n+1\}$ to represent how many message carriers are in the system (as well as a terminal state $x_k = n+1$). Additionally, we can compress the vectors y_k, z_k, e_k, w_k and u_k. For example, the state $y_k \in \mathcal{Y} \subseteq \{0,1\}^3$ can still be represented as $y_k = [y_{k,0}, y_k^m, y_k^d]^T$ as in the original problem, except that now y_k^m and y_k^d are scalars, indicating the contact events between S and a *non-carrier* node, and between a *carrier* node and an AP, respectively. Similarly, z_k, e_k, w_k and u_k can all be represented using 3-dimension vectors.

The control constraint set U_k is defined as:

$$U_k(x_k, y_k) = \begin{cases} u_k \leq y_k \\ u_k^d \leq \min\{x_k, 1\} \\ u_k^m \leq 1 - \min\{x_k, 1\} \\ c^T \cdot u_k < c_{cap} - x_k c^m \end{cases}$$

where c^T is the cost vector $[c_0, c^m, c^d]$. The constraint set is constructed in a similar fashion when compared to the original problem, and hence the detail explanation is omitted.

The system dynamics equation is the same as Eq. 7.1, whereas h_k is given by the

following:

$$h_k(x_k, u_k) = \begin{cases} n+1 & \text{if } \max\{u_{k,0}, u_k^d\} = 1 \\ x_k + u_k^m & \text{otherwise} \end{cases}$$

The probability distribution of w_k can be calculated as follows. First notice that the total contact rate for the events that can be utilized is $\lambda_{total} = \lambda_0 + (n - x_k)\lambda^m + x_k\lambda^d$. Hence we have $P(w_k = \mathbf{0}) = e^{-\lambda_{total}\delta_t}$, $P(w_{k,0} = 1) = \frac{\lambda_0}{\lambda_{total}} \cdot (1 - P(w_k = \mathbf{0}))$, $P(w_k^m = 1) = \frac{(n-x_k)\cdot\lambda^m}{\lambda_{total}} \cdot (1 - P(w_k = \mathbf{0}))$, and $P(w_k^d = 1) = \frac{x_k\cdot\lambda^d}{\lambda_{total}} \cdot (1 - P(w_k = \mathbf{0}))$.

The cost function g_k is defined as:

$$g_k(x_k, z_k, u_k) = \begin{cases} 0, & \text{if } x_k = n+1 \\ z_k^T \cdot u_k, & \text{otherwise} \end{cases}$$

And the terminal cost is defined as:

$$g_N(x_N) = \begin{cases} 0, & \text{if } x_k = n+1 \\ e_c, & \text{if } c_c + x_N \cdot c^m < c_{cap} \\ Penalty, & \text{otherwise} \end{cases}$$

With the state, system equation, cost functions defined as above, the homogeneous problem share the same objective as the original problem but with a much smaller state space. It is trivial to see that the homogeneous problem also has a threshold-based optimal policy, and the proposed Algorithm 6 can still solve the homogeneous problem. However, with the reduced state space, the time complexity for optimally solving a homogeneous problem is $O(Nn)$, since we have $|\mathcal{X}| = O(n)$ and $|\mathcal{Y}| = O(1)$.

We try to use a carefully constructed homogeneous problem to approximate the

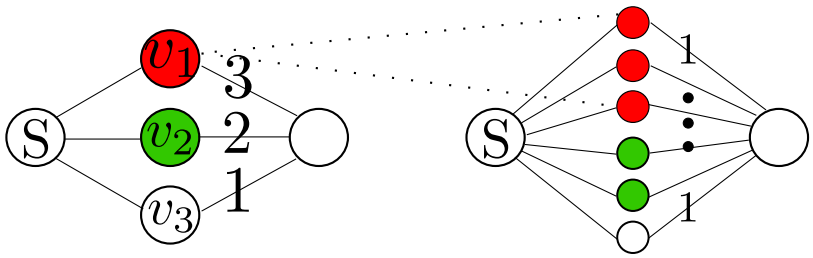


Figure 7.2: Conversion to a homogeneous problem: the left side illustrates the original problem, while the right side illustrates the converted homogeneous problem. The contact rates between v_i and APs are shown alongside the node.

original problem, and use the cost-to-go function for this simplified problem to approximate the true cost-to-go function for the original problem. The intuition is illustrated in Figure 7.2. The basic idea is to capture the offloading capability of a mobile node characterized by its contact rate to APs. Hence consider the state where a message carrier v_i has high contact rate to APs, then in the converted homogeneous problem, this corresponds to the state where more nodes are message carriers. In Figure 7.2, since v_1 has $\lambda_1^d = 3$, which is half of the total rate to APs ($\lambda_{total} = 3+2+1 = 6$), then if v_1 is a message carrier, half of the nodes (i.e., 3 nodes) in the homogeneous problem should be message carriers. We choose to convert the original problem based on the contact rates to APs due to the following reasons. First, the contact rates to APs significantly affects the offloading success rate, which has a large impact on the energy and monetary cost efficiency. Second, the source only make forwarding decisions when it is already meeting with some node. It is therefore more important to consider whether the encountered node is capable of offloading the message in the future, which is characterized by the contact rates to APs.

To construct the homogeneous problem, we have to make sure 1) the total contact rates, 2) the overall energy consumption distribution, and 3) the expected monetary

cost between S and mobile nodes and between mobile nodes and APs maintain the same when compared to the original problem. To this end, let $\tilde{n} = \lfloor \sum_i \lambda_i^d / \min_i \lambda_i^d \rfloor$ be the number of nodes in the homogeneous problem. Then we set the contact rates between S and mobile nodes, and between mobile nodes and APs as $\tilde{\lambda}^m = \sum_i \lambda_i^m / \tilde{n}$, and $\tilde{\lambda}^d = \min_i \lambda_i^d$, respectively. In this way we have $\tilde{n}\tilde{\lambda}^m = \sum_i \lambda_i^m$, and $\tilde{n}\tilde{\lambda}^d \approx \sum_i \lambda_i^d$, where the difference in the latter term comes from the rounding error for $\tilde{n}$. The expected monetary cost between S and any mobile node is $\sum_i \lambda_i^m c_i^m / \sum_j \lambda_j^m$. Hence we set $\tilde{c}^m = \sum_i \lambda_i^m c_i^m / \sum_j \lambda_j^m$. Similarly, we set $\tilde{c}^d = \sum_i \lambda_i^d c_i^d / \sum_j \lambda_j^d$. Finally, the energy consumption random variable $\tilde{e}^m$ between S and a mobile node has the CDF as $\tilde{F}^m(x) = \sum_i \lambda_i^m F_i^m(x) / \sum_j \lambda_j^m$, and the random variable $\tilde{e}^d$ between a mobile node and APs has CDF as $\tilde{F}^d(x) = \sum_i \lambda_i^d F_i^d(x) / \sum_j \lambda_j^d$.

To map the state of the original problem into the converted homogeneous problem, we have the following state conversion rules. Let $\tilde{x_k}, \tilde{y_k}$ and $\tilde{z_k}$ denote the states for the converted homogeneous problem. Then for any state x_k in the original problem, we can find its corresponding state in the homogeneous problem as $\tilde{x_k} = \sum_i x_{k,i} \lambda_i^d / \min_i \lambda_i^d$. Notice that $\tilde{x_k}$ may not be an integer. We will discuss about this later. As for y_k and z_k, we simply set the corresponding states based on the contact event. That is, set $\tilde{y}_{k,0} = y_{k,0}$; if $y_{k,i}^m = 1$ for some i then set $\tilde{y}_k^m = 1$ and $\tilde{z}_k^m = z_{k,i}^m$; if $y_{k,i}^d = 1$ for some i, set $\tilde{y}_k^d = 1$ and $\tilde{z}_k^d = z_{k,i}^d$.

7.3.3 Approximated Cost-to-go Function

We can use the cost-to-go function $\tilde{J}_k$ for the converted homogeneous problem as the approximated cost-to-go function $\hat{J}_k$ for the original problem. However, recall that $\tilde{x_k}$ might not be an integer, we have to round it to an appropriate value. To this end, let $\underline{\tilde{x_k}} = \lfloor \tilde{x_k} \rfloor$, $\overline{\tilde{x_k}} = \lceil \tilde{x_k} \rceil$, and $\rho_k = \tilde{x_k} - \underline{\tilde{x_k}}$. Then we define the approximated

cost-to-go function $\hat{J}_k$ as:

$$\hat{J}_k(x_k, y_k, z_k) = (1 - \rho_k)\tilde{J}_k(\underline{\tilde{x_k}}, \tilde{y_k}, \tilde{z_k}) + \rho_k \tilde{J}_k(\overline{\tilde{x_k}}, \tilde{y_k}, \tilde{z_k}) \tag{7.7}$$

Essentially $\hat{J}_k$ is a linear combination of $\tilde{J}_k$ evaluated at two places differentiated by whether $\tilde{x_k}$ is round up or down, weighted by the distance between $\tilde{x_k}$ and the nearest smaller integer. Hence if the precise value of $\tilde{x_k}$ is close to any of the integer, then $\hat{J}_k$ will be close to the corresponding value of $\tilde{J}_k$; otherwise, we approximate it using the linear interpolation.

7.3.4 One Step Lookahead Policy

Now that we have presented the conversion of the original problem to a homogeneous problem, and the approximated cost-to-go function, we present the construction of the CEO policy as follows:

1) Convert the original problem into a homogeneous problem using the rules presented in Section 7.3.2.

2) Solve the converted homogeneous problem optimally using Algorithm 6 to get the true cost-to-go function $\tilde{J}_k$ for the homogeneous problem.

3) At any time slot k and for a given state (x_k, y_k, z_k), determine the decision by solving the minimization problem defined in Eq. 7.6, where the state conversion rules are defined in Section 7.3.2 and the approximated cost-to-go function $\hat{J}_k$ is defined in Eq. 7.7.

Notice that in *Step 3)*, we can still utilize Eq. 7.4 to calculate the threshold, except that we need to replace the true cost-to-go function with the approximated one.

By using the *one-step lookahead policy*, we avoid the calculation of the thresholds for *every* possible states, but instead only calculate the threshold whenever a

particular state is observed. Although we still need to calculate the true cost-to-go functions of the converted homogeneous problem in order to construct the approximated cost-to-go function, such computation has significantly lower complexity.

7.3.5 CEO Data Offloading Protocol

We present the CEO data offloading protocol as follows:

First, a mobile node needs to maintain the contact rates, energy consumption estimation and monetary cost to other mobile nodes and APs it encountered, and exchanges such information when it encounters another mobile node. In this way, each node can maintain the necessary information for constructing policies.

Second, when a new message is generated, the node calculates the CEO policy based on Section 7.3.4, and maintains the state x_k for this message.

Third, when it encounters a mobile nodes or AP, transfer the message if the estimated energy consumption is below the policy threshold. If the forwarding is to a mobile nodes, update the state x_k. Additionally, transfer the thresholds for x_k to the encountered mobile node which is used for forwarding to APs.

7.4 Performance Evaluation

In this section, we throughly evaluate the performance of our CEO policy through trace-driven simulations using the Opportunistic Network Environment (ONE) [42] simulator.

7.4.1 Simulation Setup

We use the *Kaist* [73] mobility trace as the input to ONE. The *Kaist* trace contains daily GPS logs of 92 participants moving around the university campus of KAIST, Korea. We deploy n_{ap} virtual APs in the area with the most user mobilities covered by the coordinates (-1000,-1000) to (1000,1000) as have been similarly done

in [35]. We set the n_{ap} to 50 and 200 to simulate sparse and dense AP deployments, respectively. We assume that both the mobile nodes and APs have communication range of 100m.

The bandwidth for both WiFi and cellular interface is set to 2Mbps. Messages size is set to 1MB. The source is selected randomly from the 92 nodes. The message interval is uniformly distributed in the interval of $[250, 300]$ seconds. The monetary cost for transmitting a message using WiFi interface is set to 0, while using cellular interface is set to 10. We set the monetary cost constraint c_{cap} for transmitting a message to be 5 and 15, indicating whether or not the node would like to use cellular communication. If the user set c_{cap} to 5, it means that the user would like to avoid the cellular usage; otherwise, the user can accept cellular communication.

We set each node v_i's transmission and reception *energy per bit* using WiFi interface to be Gaussian random variables $N(E_i, E_i/2)$, where E_i is the expectation. E_i is in turn derived from another Gaussian random variable $N(1, 0.5)$ $\mu J/bit$. Notice that *energy per bit* can be estimated using information about power, data rate, signal strength [68] [75] [32]. We set the mean of the *energy per bit* to be 1 $\mu J/bit$ as it falls into the interval based on real world measurements [32]. We set the standard deviation to be 0.5 in order to have a relatively large support, as it was reported in [75] that the *energy per bit* can be up to 6 times large when the node has poor link quality compared to good link quality. The ratio between cellular and WiFi uplink energy consumption ranges from 2.53 to an order of magnitude [32]. To this end, we set the *energy per bit* for cellular communication to be 5 $\mu J/bit$, which falls in the aforementioned interval. We also vary the cellular energy consumption to evaluate the performance in a wider spectrum.

We compare CEO to the following three strategies:

WiFi Naive (WN): This strategy will transmit the message when the node

encounters any AP, without considering the energy consumption and potential future contacts.

WiFi Only (WO): This strategy is essentially a special case for CEO. The policy is constructed using the same approach as CEO, except that all mobile nodes are ignored (and hence the name *WiFi only*). We compare to WO to demonstrate the importance of considering mobile nodes when offloading.

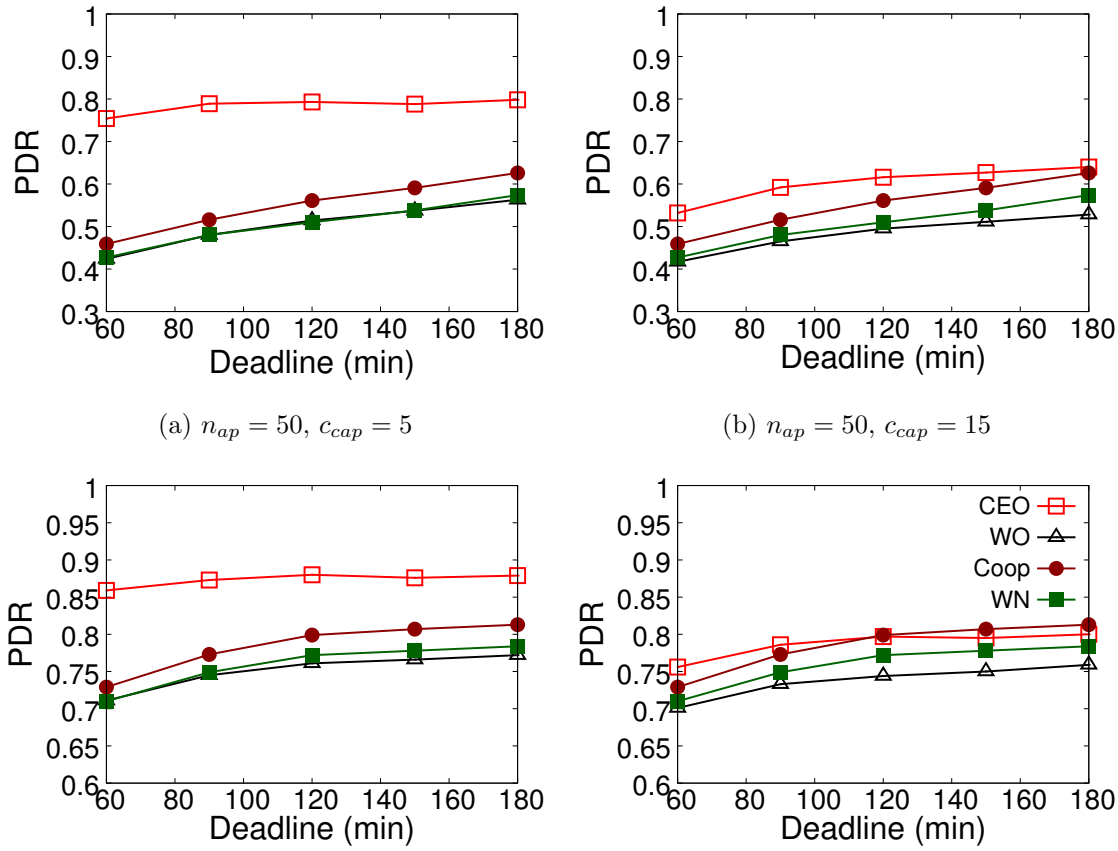
Cooperative Offloading (Coop) [53]: Coop is a two-hop based forwarding strategy which makes the forwarding decision based on the path's *availability probability* and *delivery probability*. We compare to Coop to demonstrate the importance of considering energy consumption.

For each scenario, we vary the following two parameters, *deadline* and *cellular energy consumption*. Deadline of the messages varies from 60 minutes up to 180 minutes. As for *cellular energy consumption*, we vary the *energy per bit* from 2.5 $\mu J/bit$ up to 10 $\mu J/bit$. When we vary the cellular energy consumption, we set the deadline to 120 minutes.

We are interested in the following two metrics: *packet delivery ratio* (PDR) and *energy consumption*. PDR is defined as the proportion of the messages that are offloaded through APs, i.e., delivered to one of the APs before deadline. PDR reflects how much monetary cost can be saved for the user. Energy consumption is the total energy consumption for transmitting the messages, including the energy consumption incurred on both WiFi and cellular interfaces. Each data point is averaged over 30 simulation runs for statistic significance.

7.4.2 Evaluation When Varying Deadline

Figures 7.3 and 7.4 show the results when we vary the deadline from 60 minutes up to 180 minutes.



(a) $n_{ap} = 50$, $c_{cap} = 5$

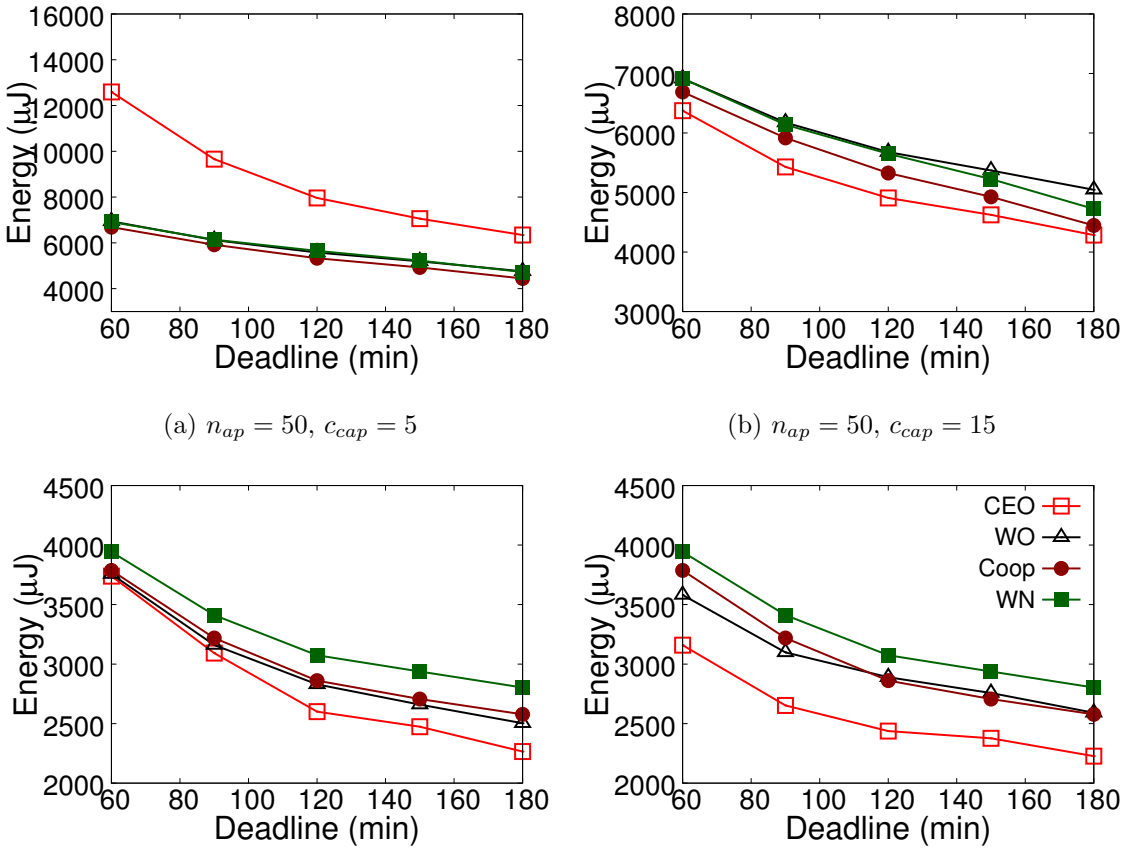
(b) $n_{ap} = 50$, $c_{cap} = 15$

(c) $n_{ap} = 200$, $c_{cap} = 5$

(d) $n_{ap} = 200$, $c_{cap} = 15$

Figure 7.3: PDR results for varying deadline from 60 min up to 180 min

Figures 7.3a 7.3b 7.4a and 7.4b show the results when the AP density is sparse. First, we notice that PDR increases when the deadline increases for all policies, which is intuitive. However, the increase of PDR for CEO is smaller when compared to other policies. For example, in Figure 7.3a the increase of PDR for CEO and Coop from 60 min to 180 min are 4.4% and 16.7%, respectively. This is because when the deadline becomes longer, the energy threshold of CEO becomes smaller, which means CEO will look for more energy efficient transfer opportunities. Hence the reduction of energy consumption with the increase of deadline is not only due to the increase



(a) $n_{ap} = 50$, $c_{cap} = 5$

(b) $n_{ap} = 50$, $c_{cap} = 15$

(c) $n_{ap} = 200$, $c_{cap} = 5$

(d) $n_{ap} = 200$, $c_{cap} = 15$

Figure 7.4: Energy consumption results for varying deadline from 60 min up to 180 min

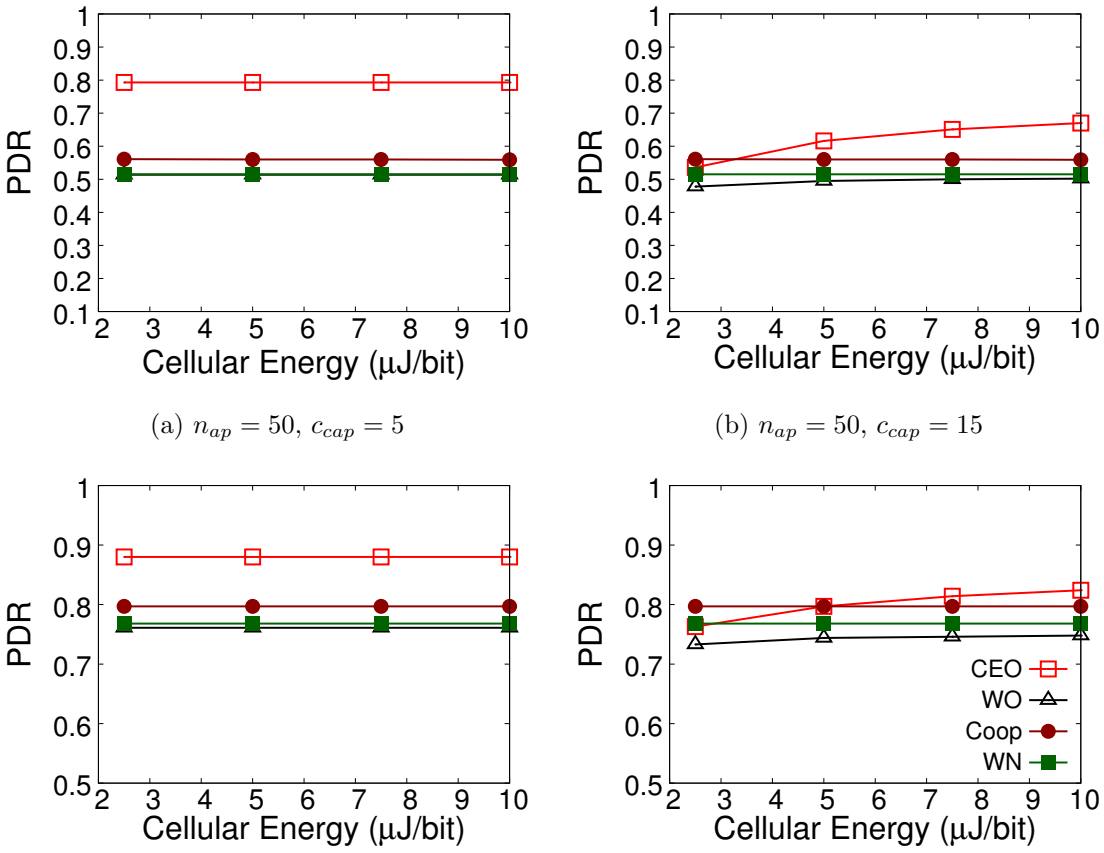
of PDR, but more importantly due to CEO's ability of finding more energy efficient transfer opportunities. Second, if the user would like to avoid cellular network usage, i.e., when $c_{cap} = 5$, we see that CEO offers significantly better performance in terms of PDR, with up to 64.2% increase of PDR compared to the second best policy. This can save a significant amount of monetary cost for the users as more traffic is offloaded. However, this also incurs a large energy consumption as we can see in Figure 7.4a. On the other hand, if the user can afford the monetary cost for cellular, i.e., $c_{cap} = 15$, then CEO is less aggressive in terms of trying to offload the

traffic but be patient and tries to find better transfer opportunities. As we can see in Figure 7.3b and 7.4b, the increase of PDR for CEO is up to 15.9%, but with up to 8.2% energy consumption reduction when compared to Coop.

When the AP density is high (Figures 7.3c 7.3d 7.4c and 7.4d), mobile nodes have higher chance to meet with APs within the deadline. In this case, if the user would like to avoid cellular communication, i.e., $c_{cap} = 5$, then CEO can improve the PDR up to 17.8% when compared to the second best policy Coop. In terms of energy consumption, CEO achieves up to 12.1% reduction. The increase of PDR is mainly due to CEO's ability of utilizing other mobile nodes and message replication. This, however, does not incur large amount of additional energy consumption which is due to CEO's ability of filtering poor transfer opportunities that are not energy efficient. On the other hand, if the user would like to afford cellular monetary cost, i.e., $c_{cap} = 15$, then CEO achieves comparable PDR compared to other policies, but with 12.1% to 17.5% energy consumption reduction when compared to the Coop policy.

7.4.3 Evaluation When Varying Cellular Energy Consumption

Figures 7.5 and 7.6 show the results when we vary the cellular energy per bit c_e from 2.5 up to 10 $\mu J/bit$. When the AP density is sparse (Figures 7.5a 7.5b 7.6a and 7.6b), if $c_{cap} = 5$ then CEO can achieve an average of 41.6% increase of PDR compared to Coop, while also incurs higher energy consumption. However, notice that when the c_e increases, CEO's energy consumption becomes comparable to Coop policy. If $c_{cap} = 15$, then CEO achieves PDR ranging from 4.4% less when $c_e = 2.5$ to 19.8% more when $c_e = 10$ compared to Coop. Meanwhile, CEO also reduces energy consumption by 4.0% to 15.9%. We notice that the PDR of CEO actually increases from 53.6% to 67.0% with the increase of c_e. This is because the increase of c_e will



(a) $n_{ap} = 50$, $c_{cap} = 5$

(b) $n_{ap} = 50$, $c_{cap} = 15$

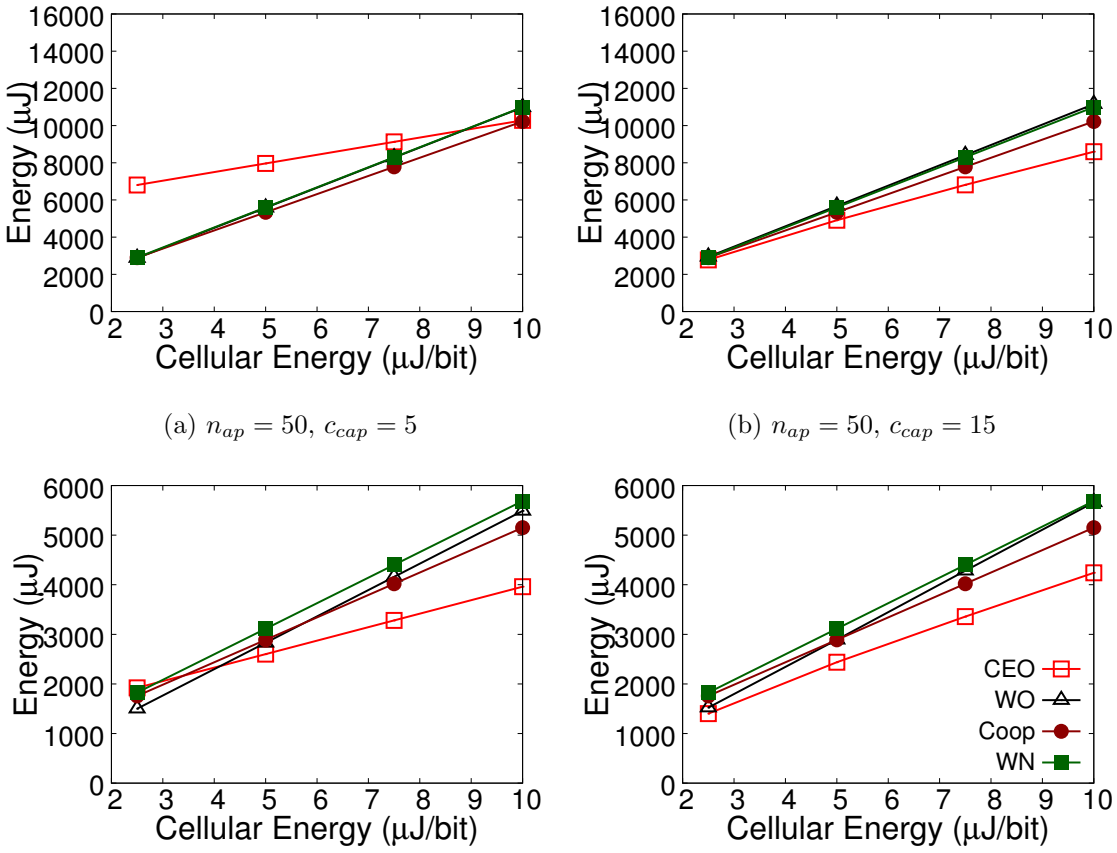
(c) $n_{ap} = 200$, $c_{cap} = 5$

(d) $n_{ap} = 200$, $c_{cap} = 15$

Figure 7.5: PDR results with different cellular energy consumption

incur higher terminal cost. Consequently, the derived policy will be more incline to the usage of WiFi and other mobile nodes as they are more energy efficient by increasing the energy threshold and accepting more transfer opportunities.

When the AP density is high, if $c_{cap} = 5$, then CEO achieves an average of 10.4% increase of PDR with energy consumption ranges from 9.3% more when $c_e = 2.5$ to 21.3% less when $c_e = 10$, compared to Coop. When $c_{cap} = 15$, we observe a similar pattern, i.e., the PDR of CEO increases from 76.3% to 82.4% when c_e increases from 2.5 to 10. When compared to Coop, CEO has comparable PDR, but reduces energy



(a) $n_{ap} = 50$, $c_{cap} = 5$

(b) $n_{ap} = 50$, $c_{cap} = 15$

(c) $n_{ap} = 200$, $c_{cap} = 5$

(d) $n_{ap} = 200$, $c_{cap} = 15$

Figure 7.6: Energy consumption results with different cellular energy consumption

consumption by 15.6% to 20.3%.

7.4.4 Summary of Evaluation Results

In general, we see that CEO can adapt well to the users' preference of monetary cost and the environments. If the user would like to avoid the cellular monetary cost, then CEO can achieve significantly higher PDR with reasonable increase or even comparable energy consumption depending on the density of AP deployments. If the user can afford the cellular monetary cost, then CEO becomes more patient in terms of finding more energy efficient transfer opportunities. And hence CEO

achieves comparable or better PDR, but with more energy consumption reduction. In addition, cellular energy consumption may also have an impact on the offloading performance. When cellular energy consumption increases, it forces CEO to offload more traffic as it may be more energy efficient to utilize poor quality WiFi or message replication than to use the energy inefficient cellular communication.

8. CONCLUSION AND FUTURE WORK

In this section, we conclude this dissertation and present the future work.

8.1 Conclusion

With the prevalence of mobile devices nowadays and the trend of continuously increase in the future, Opportunistic Mobile Network may play a critical role in future mobile networks. In contrast to previous assumptions, OMN users actually experience diverse connectivity: the connectivity may range from sparsely connected to well connected and the network may coexist with infrastructure. In this dissertation, we propose a communication framework that consists of a series of algorithms and protocols which provide *energy efficient*, *robust* and *cost-aware* communication services to OMN applications. To tackle the problem, we carefully move along the connectivity spectrum: from sparsely connected to well connected, and investigate the inefficiencies of existing protocols.

When the connectivity is sparse, through the analysis of both real world and synthetic mobility traces, we show that multi-node contacts are present and that they are stable in terms of duration, which is in sharp contrast to the traditional pairwise contact model assumption. Through real world experiments on routers with state-of-art implementation of OMN routing protocols, we further show that the existing protocols may have suboptimal behaviors such as redundant transmissions. Consequently, we develop and present a Multi-node Contact Optimization algorithm which receives input from routing protocols and produces a packet transmission schedule, allowing nodes transmitting network coded packets to better utilize multi-node contact opportunities. We demonstrate through simulations using real world mobility traces that our algorithm is able to reduce the number of transmissions

while maintaining routing performance of packet delivery rate and packet delivery delay.

We demonstrate the existence of the energy consumption imbalance problem through both theoretical and empirical analysis for social based OMN routing protocols, which intends to reduce the packet replication and transmissions through utilizing socially popular nodes. We propose a new social based routing metric and a distributed algorithm for mobile nodes to obtains the metric. The new metric explicitly minimizes the memory load imbalance and thereby reduces the energy consumption imbalance. We further propose an intra-TCC routing mechanism to avoid the over-utilization nodes within a TCC. Enabled by these tools, we propose the Energy-balanced Routing protocol. We analyze the proposed protocol through theoretical analysis and show that the average energy consumptions for all mobile nodes converge to the same value after sufficient amount of time. Through extensive simulations, we show that our protocol achieves a more balanced energy consumption while maintaining comparable routing performance.

For networks where the connectivity may range from sparsely connected to well connected temporally or spatially, we propose the Hybrid Routing Protocol. We carefully study the benefit of packet replication, i.e., the key structural difference between routing protocols for well connected and sparsely connected networks. We demonstrate, for the first time, that delay correlation may significantly impact the replication gain. We propose a novel Joint ICT model to capture ICT correlation among different nodes which enable us to design an efficient forwarding strategy. We design a regret-minimization based algorithm that is able to dynamically adjust the amount of replication in order to deal with any dynamic network environments. We implement a prototype of HRP and deploy it on wireless routers to evaluate its performance. We further conduct extensive simulation study to demonstrate HRP's

superior performance when compared to state-of-art hybrid routing protocols and OMN routing protocols.

For networks which coexist with infrastructure, i.e., with continuous cellular network coverage and intermittent WiFi access points coverage, we propose a Cost-aware Energy-efficient Data Offloading protocol. We formulate the problem, where both mobile-to-infrastructure and mobile-to-mobile contacts are considered, as a discrete time optimal control problem. We show that the optimal policy for such a problem has a threshold-based structure. Due to the curse of dimensionality, the algorithm for constructing the optimal policy runs in exponential time. Therefore we further propose an approximation based CEO policy which can be calculated efficiently. The approximation algorithm is derived from a one-step look forward policy which uses the optimal policy for a carefully constructed homogeneous problem that approximates the original heterogeneous problem. Through extensive simulation, we show that CEO adapts well to the users' preference of monetary cost and the environments, and outperform state-of-art protocols.

8.2 Future Work

In this section, we present a few ideas for future work.

8.2.1 Improving Multi-node Contact Optimization

Currently, our MCO algorithm is centralized, i.e., one master node is in charge of collecting routing decisions and calculating the transmission schedule. In order to make it more scalable and robust, one interesting topic is to design a distributed version of the Multi-node Contact Optimization algorithm. Additionally, it would be interesting to explore how the increased capacity for each contact may impact the design of routing protocols. With our contact optimization algorithm, the capacity of each contact is actually increased since the number of transmission is reduced for

exchanging the same amount of information. A natural next step is to explore the design of a "network coding aware" routing protocol which may further improve the routing performance.

8.2.2 Mobility Correlation Modeling

In this dissertation, we propose to use the Multi-variate Exponential distribution to model the Joint Inter-Contact Times among mobile nodes. The MVE distribution exhibits a few nice properties: its underlying random process naturally explains the reason of mobility correlation, i.e., co-location of mobile nodes; its marginal distribution remains exponential which echoes many existing research on inter-contact time distribution; it is elegant and mathematical tractable.

There are a few future directions to better characterize mobility correlation. First, it is interesting to explore other multi-dimensional distributions that have different types of marginal distributions. There are still doubts in the literature about what distribution best describes the pairwise inter-contact times, where some work argue that long tail distributions, or long tail distribution with exponential cutoff are better candidates. It is therefore also important to investigate joint inter-contact times distributions that generates these marginal distributions. Second, it is interesting to explore other inter-contact time correlation structures. The MVE distribution is used to capture positive correlation. A natural next step questions is: "can inter-contact times also exhibit negative correlation and what model is appropriate to capture such correlations?". Additionally, MVE only captures the correlation resulted from node co-location: when a node meets with two other nodes at the same time. It is interesting to explore other types of correlation, e.g., two nodes share a similar schedule but the schedules are shifted in time.

8.2.3 Energy-optimal Data Offloading

Another topic that deserves further exploring is the energy-optimal data offloading problem. In this dissertation, we propose a two-hop based cost-aware energy efficient offloading strategy, where the mobile nodes which receive the message only deliver it to the destination. One potential direction for future exploration is to generalize the "two-hop" forwarding scheme, where we allow message carriers to also forward (or replicate) to other mobile nodes. This is, however, challenging as we have already seen that the optimal solution to the "two-hop" based offloading problem is intractable. We envision that a carefully designed approximation based algorithm is needed.

Yet another potential direction is to incorporate the queueing effects in the design of the offloading strategy. Currently, the system state does not include the queue length at each mobile node. However, intuitively, if the mobile node is already holding many messages, the energy threshold might be higher in order to quickly forward the message to avoid reaching deadline. It is important to consider this aspect, as the bandwidth or the contact duration might not be sufficient to transmit all the messages during each contact.

8.2.4 System Implementation

Except for HRP where we have a proof-of-concept implementation and evaluation on wireless testbeds, we mostly use simulators with mobility datasets to study the performance of the proposed protocols. It is therefore important to implement the proposed protocols in the actual systems in order to understand the performance and overhead in real world scenarios.

REFERENCES

[1] Mikrotik router. http://www.mikrotik.com/.

[2] Cisco visual networking index: Global mobile data traffic forecast update, 2016 to 2021. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, 2017.

[3] B. C. Arnold. Parameter estimation for a multivariate exponential distribution. *Journal of the American Statistical Association*, 63(323):848–852, 1968.

[4] A. Balasubramanian, B. Levine, and A. Venkataramani. Dtn routing as a resource allocation problem. In *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '07, pages 373–384, New York, NY, USA, 2007. ACM.

[5] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3g using wifi. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 209–222, New York, NY, USA, 2010. ACM.

[6] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine. Relays, base stations, and meshes: Enhancing mobile networks with infrastructure. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, pages 81–91, New York, NY, USA, 2008. ACM.

[7] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.

[8] E. Bulut, S. C. Geyik, and B. K. Szymanski. Utilizing correlated node mobility for efficient dtn routing. *Pervasive Mob. Comput.*, 13:150–163, Aug. 2014.

[9] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pages 1–11, April 2006.

[10] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss. RFC 4838, Delay-Tolerant Networking Architecture. *IRTF DTN Research Group*, 2007.

[11] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, June 2007.

[12] M. A. R. Chaudhry and A. Sprintson. Efficient algorithms for index coding. In *IEEE INFOCOM Workshops 2008*, pages 1–4, April 2008.

[13] H. Chenji, W. Zhang, R. Stoleru, and C. Arnett. Distressnet: A disaster response system providing constant availability cloud-like services. *Ad Hoc Networks*, 11(8):2440 – 2460, 2013.

[14] F. C. Commission. Hurricane maria communications status report for sept. 29. https://www.fcc.gov/document/hurricane-maria-communications-status-report-sept-21, 2017.

[15] M. Conti and M. Kumar. Opportunities in opportunistic computing. *Computer*, 43(1):42–50, Jan 2010.

[16] J. Cook. Hong kong protesters are communicating via a mobile app that doesn't actually use the internet. http://www.businessinsider.com/firechat-the-

app-hong-kong-uses-to-organize-protests-2014-9, Sept. 2014.

[17] E. M. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '07, pages 32–40, New York, NY, USA, 2007. ACM.

[18] L. Delosires and S. Nadjm-Tehrani. Batman store-and-forward: The best of the two worlds. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 721–727, March 2012.

[19] M. Demmer and K. Fall. Dtlsr: Delay tolerant routing for developing regions. In *Proceedings of the 2007 Workshop on Networked Systems for Developing Regions*, NSDR '07, pages 5:1–5:6, New York, NY, USA, 2007. ACM.

[20] N. Ding, D. Wagner, X. Chen, A. Pathak, Y. C. Hu, and A. Rice. Characterizing and modeling the impact of wireless signal strength on smartphone battery drain. In *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '13, pages 29–40, New York, NY, USA, 2013. ACM.

[21] M. Doering, S. Lahde, J. Morgenroth, and L. Wolf. Ibr-dtn: An efficient implementation for embedded systems. In *Proceedings of the Third ACM Workshop on Challenged Networks*, CHANTS '08, pages 117–120, New York, NY, USA, 2008. ACM.

[22] N. Eagle and A. (Sandy) Pentland. Reality mining: Sensing complex social systems. *Personal Ubiquitous Comput.*, 10(4):255–268, Mar. 2006.

[23] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot. Delegation forwarding. In *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc*

Networking and Computing, MobiHoc '08, pages 251–260, New York, NY, USA, 2008. ACM.

[24] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 27–34, New York, NY, USA, 2003. ACM.

[25] R. K. Ganti, F. Ye, and H. Lei. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11):32–39, November 2011.

[26] G. Gao, M. Xiao, J. Wu, K. Han, and L. Huang. Deadline-sensitive mobile data offloading via opportunistic communications. In *2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–9, June 2016.

[27] W. Gao, G. Cao, T. L. Porta, and J. Han. On exploiting transient social contact patterns for data forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 12(1):151–165, Jan 2013.

[28] W. Gao, Q. Li, and G. Cao. Forwarding redundancy in opportunistic mobile networks: Investigation and elimination. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 2301–2309, April 2014.

[29] W. Gao, Q. Li, B. Zhao, and G. Cao. Multicasting in delay tolerant networks: A social network perspective. In *Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '09, pages 299–308, New York, NY, USA, 2009. ACM.

[30] S. Ha, S. Sen, C. Joe-Wong, Y. Im, and M. Chiang. Tube: Time-dependent pricing for mobile data. In *Proceedings of the ACM SIGCOMM 2012 Confer-*

ence on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '12, pages 247–258, New York, NY, USA, 2012. ACM.

[31] B. Han, P. Hui, V. S. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *IEEE Transactions on Mobile Computing*, 11(5):821–834, May 2012.

[32] 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 *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 225–238, New York, NY, USA, 2012. ACM.

[33] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, pages 244–251, New York, NY, USA, 2005. ACM.

[34] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: Social-based forwarding in delay tolerant networks. In *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '08, pages 241–250, New York, NY, USA, 2008. ACM.

[35] P. Hui, A. Lindgren, and J. Crowcroft. Empirical evaluation of hybrid opportunistic networks. In *2009 First International Communication Systems and Networks and Workshops*, pages 1–10, Jan 2009.

[36] Y. Im, C. Joe-Wong, S. Ha, S. Sen, T. . Kwon, and M. Chiang. Amuse: Empowering users for cost-aware offloading with throughput-delay tradeoffs. *IEEE Transactions on Mobile Computing*, 15(5):1062–1076, May 2016.

[37] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized link state routing protocol for ad hoc networks. In *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century.*, pages 62–68, 2001.

[38] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized link state routing protocol for ad hoc networks. In *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century.*, pages 62–68, 2001.

[39] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '04, pages 145–158, New York, NY, USA, 2004. ACM.

[40] E. P. C. Jones, L. Li, J. K. Schmidtke, and P. A. S. Ward. Practical routing in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 6(8):943–959, Aug 2007.

[41] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. Xors in the air: Practical wireless network coding. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '06, pages 243–254, New York, NY, USA, 2006. ACM.

[42] A. Keränen, J. Ott, and T. Kärkkäinen. The one simulator for dtn protocol evaluation. In *Proceedings of the 2Nd International Conference on Simulation Tools and Techniques*, Simutools '09, pages 55:1–55:10, ICST, Brussels, Belgium, Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[43] M. Khouzani, S. Eshghi, S. Sarkar, N. B. Shroff, and S. S. Venkatesh. Optimal energy-aware epidemic routing in dtns. In *Proceedings of the Thirteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '12, pages 175–182, New York, NY, USA, 2012. ACM.

[44] C. Kretschmer, S. Ruhrup, and C. Schindelhauer. Dt-dymo: Delay-tolerant dynamic manet on-demand routing. In *2009 29th IEEE International Conference on Distributed Computing Systems Workshops*, pages 493–498, June 2009.

[45] J. Lakkakorpi, M. Pitkänen, and J. Ott. Adaptive routing in mobile opportunistic networks. In *Proceedings of the 13th ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, MSWIM '10, pages 101–109, New York, NY, USA, 2010. ACM.

[46] J. Lakkakorpi, M. Pitkänen, and J. Ott. Using buffer space advertisements to avoid congestion in mobile opportunistic dtns. In *Proceedings of the 9th IFIP TC 6 International Conference on Wired/Wireless Internet Communications*, WWIC'11, pages 386–397, Berlin, Heidelberg, 2011. Springer-Verlag.

[47] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong. Mobile data offloading: How much can wifi deliver? In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 26:1–26:12, New York, NY, USA, 2010. ACM.

[48] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, July 2003.

[49] S. Liu and A. D. Striegel. Exploring the potential in practice for opportunistic networks amongst smart mobile devices. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, MobiCom '13, pages 315–326, New York, NY, USA, 2013. ACM.

[50] Y. Liu, D. R. Bild, D. Adrian, G. Singh, R. P. Dick, D. S. Wallach, and Z. M. Mao. Performance and energy consumption analysis of a delay-tolerant network for censorship-resistant communication. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '15, pages 257–266, New York, NY, USA, 2015. ACM.

[51] Z. Lu, G. Cao, and T. L. Porta. Networking smartphones for disaster recovery. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9, March 2016.

[52] Z. Lu, X. Sun, and T. L. Porta. Cooperative data offloading in opportunistic mobile networks. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9, April 2016.

[53] Z. Lu, X. Sun, and T. L. Porta. Cooperative data offloading in opportunistic mobile networks. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9, April 2016.

[54] A. W. Marshall and I. Olkin. A multivariate exponential distribution. *Journal of the American Statistical Association*, 62(317):30–44, 1967.

[55] M. McNett and G. M. Voelker. Access and mobility of wireless pda users. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(2):40–55, Apr. 2005.

[56] F. Mehmeti and T. Spyropoulos. Is it worth to be patient? analysis and optimization of delayed mobile data offloading. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 2364–2372, April 2014.

[57] D. G. Murray, E. Yoneki, J. Crowcroft, and S. Hand. The case for crowd computing. In *Proceedings of the Second ACM SIGCOMM Workshop on Networking,*

Systems, and Applications on Mobile Handhelds, MobiHeld '10, pages 39–44, New York, NY, USA, 2010. ACM.

[58] M. Musolesi and C. Mascolo. Car: Context-aware adaptive routing for delay-tolerant mobile networks. *IEEE Transactions on Mobile Computing*, 8(2):246–260, Feb 2009.

[59] S. C. Nelson, M. Bakht, and R. Kravets. Encounter-based routing in dtns. In *IEEE INFOCOM 2009*, pages 846–854, April 2009.

[60] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[61] J. Ott, D. Kutscher, and C. Dwertmann. Integrating dtn and manet routing. In *Proceedings of the 2006 SIGCOMM Workshop on Challenged Networks*, CHANTS '06, pages 221–228, New York, NY, USA, 2006. ACM.

[62] A. Passarella and M. Conti. Analysis of individual pair and aggregate inter-contact times in heterogeneous opportunistic networks. *IEEE Transactions on Mobile Computing*, 12(12):2483–2495, Dec 2013.

[63] O. Pearce, T. Gamblin, B. R. de Supinski, M. Schulz, and N. M. Amato. Quantifying the effectiveness of load balance algorithms. In *Proceedings of the 26th ACM International Conference on Supercomputing*, ICS '12, pages 185–194, New York, NY, USA, 2012. ACM.

[64] L. Pelusi, A. Passarella, and M. Conti. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*, 44(11):134–141, November 2006.

[65] A. Picu, T. Spyropoulos, and T. Hossmann. An analysis of the information spreading delay in heterogeneous mobility dtns. In *2012 IEEE International*

Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), pages 1–10, June 2012.

[66] A.-K. Pietiläinen and C. Diot. Dissemination in opportunistic social networks: The role of temporal communities. In *Proceedings of the Thirteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '12, pages 165–174, New York, NY, USA, 2012. ACM.

[67] J. M. Pujol, A. L. Toledo, and P. Rodriguez. Fair routing in delay tolerant networks. In *IEEE INFOCOM 2009*, pages 837–845, April 2009.

[68] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. Energy-delay tradeoffs in smartphone applications. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 255–270, New York, NY, USA, 2010. ACM.

[69] M. Radenkovic and A. Grundy. Efficient and adaptive congestion control for heterogeneous delay-tolerant networks. *Ad Hoc Networks*, 10(7):1322 – 1345, 2012.

[70] C. Raffelsberger and H. Hellwagner. A hybrid manet-dtn routing scheme for emergency response scenarios. In *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 505–510, March 2013.

[71] F. Rebecchi, M. D. de Amorim, V. Conan, A. Passarella, R. Bruno, and M. Conti. Data offloading techniques in cellular networks: A survey. *IEEE Communications Surveys Tutorials*, 17(2):580–603, Secondquarter 2015.

[72] S. I. Resnick. *Adventures in Stochastic Processes*. Birkhauser Verlag, Basel, Switzerland, Switzerland, 1992.

[73] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong. CRAWDAD dataset ncsu/mobilitymodels (v. 2009-07-23). Downloaded from https://crawdad.org/ncsu/mobilitymodels/20090723, July 2009.

[74] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong. On the levy-walk nature of human mobility. *IEEE/ACM Transactions on Networking*, 19(3):630–643, June 2011.

[75] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan. Bartendr: A practical approach to energy-aware cellular data scheduling. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking*, MobiCom '10, pages 85–96, New York, NY, USA, 2010. ACM.

[76] M. R. Schurgot, C. Comaniciu, and K. Jaffres-Runser. Beyond traditional dtn routing: social networks for opportunistic communication. *IEEE Communications Magazine*, 50(7):155–162, July 2012.

[77] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD dataset cambridge/haggle (v. 2009-05-29). Downloaded from https://crawdad.org/cambridge/haggle/20090529, May 2009.

[78] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang. Incentivizing time-shifting of data: a survey of time-dependent pricing for internet access. *IEEE Communications Magazine*, 50(11):91–99, November 2012.

[79] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, pages 252–259, New York, NY, USA, 2005. ACM.

[80] X. Tie, A. Venkataramani, and A. Balasubramanian. R3: Robust replication routing in wireless networks with diverse connectivity characteristics. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, pages 181–192, New York, NY, USA, 2011. ACM.

[81] A. Tonnesen. Implementing and extending the optimized link state routing protocol. Master's thesis, University of Oslo, Norway, 2004.

[82] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical report, Duke University, 2000.

[83] A. Vulimiri, P. B. Godfrey, R. Mittal, J. Sherry, S. Ratnasamy, and S. Shenker. Low latency via redundancy. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 283–294, New York, NY, USA, 2013. ACM.

[84] J. Whitbeck and V. Conan. Hymad: Hybrid dtn-manet routing for dense and highly dynamic wireless networks. *Comput. Commun.*, 33(13):1483–1492, Aug. 2010.

[85] M. Xiao, J. Wu, and L. Huang. Community-aware opportunistic routing in mobile social networks. *IEEE Transactions on Computers*, 63(7):1682–1695, July 2014.

[86] C. Yang and R. Stoleru. Routing protocol-independent contact optimization for opportunistic social networks. In *2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 534–541, Oct 2014.

[87] C. Yang and R. Stoleru. On balancing the energy consumption of routing protocols for opportunistic social networks. In *2015 IEEE 34th International Perfor-*

mance Computing and Communications Conference (IPCCC), pages 1–9, Dec 2015.

[88] C. Yang and R. Stoleru. On balancing the energy consumption of routing protocols for opportunistic social networks. Technical report, Texas A&M University, 2015.

[89] C. Yang and R. Stoleru. Hybrid routing in wireless networks with diverse connectivity. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '16, pages 71–80, New York, NY, USA, 2016. ACM.

[90] J. Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.

[91] X. Zhang and G. Cao. Transient community detection and its application to data forwarding in delay tolerant networks. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*, pages 1–10, Oct 2013.

[92] X. Zhang and G. Cao. Efficient data forwarding in mobile social networks with diverse connectivity characteristics. In *2014 IEEE 34th International Conference on Distributed Computing Systems*, pages 31–40, June 2014.

[93] H. Zhu, L. Fu, G. Xue, Y. Zhu, M. Li, and L. M. Ni. Recognizing exponential inter-contact time in vanets. In *2010 Proceedings IEEE INFOCOM*, pages 1–5, March 2010.