EFFECTIVE COLLABORATION IN OPPORTUNITIC NETWORKS

by

UMAIR SADIQ

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2013

To my *ammi jee*.

ACKNOWLEDGEMENTS

The research work presented in this thesis was carried out with support from the the US National Science Foundation under grants ECCS-0824120, CSR 0834493, and CNS-1117866.

I express my gratitude to my PhD advisor, Prof. Mohan Kumar, for his extensive guidance and support during my studies. His disciplined work ethic, attention to detail, and dedication to teach and work with his students has been a great motivation and learning experience for me. I thank him for being an inspiration for me as an advisor and for gradually enabling me to become an independent researcher.

I thank the professors on my committee: Prof. Manfred Huber for discussions on mechanism design for multiagent systems, Prof. Gautam Das for lectures on randomized algorithms and approximation schemes, Prof. Matthew Wright for sharing key insights on incentive schemes for pervasive security and Prof. Yonghe Liu for discussions on modeling opportunistic networks.

I thank Dr Andrea Passarella and Dr Marco Conti for numerous critical reviews of my research work on service composition - their guidance has been extremely useful in design, implementation and analysis of simulation experiments. I thank Prof. Chris Ding for discussions on social and random networks. I also thank graduate advisors, Profs. Bahram Khalili and Ramez Elmasri for their dedication in guiding graduate students through each milestone during the course of study. I thank my collegues at The University of Texas at Arlington for their support.

I thank my friends - Wajahat, Waseem, Fawaz, Phillip, Stephanie, Gaba, Theresa, Shiraz, Michael, Derek, Sunny, Shamyl, Mughees, and Wahab. I am deeply

indebted to you all – thank you so much for so many pleasant and delightful memories, thank you for your care and support, thank you for being my friend.

I thank my family: my dear wife Sidra for making me a better person and for helping me throughout my studies, my father who is my role model of forbearance, my mother who cares for me more than I care for my own self, my sister Ammara who always guides me with patience, my brother Bilal who teaches and corrects me in so many ways, and finally my Uncle Farooq, my mamoo jee, for his immense devotion, love and care – it means the whole wide world to me.

April 1, 2013

ABSTRACT

EFFECTIVE COLLABORATION IN OPPORTUNITIC NETWORKS

UMAIR SADIQ, Ph.D.

The University of Texas at Arlington, 2013

Supervising Professor: Mohan Kumar

Opportunistic networks formed by users' mobile devices have the potential to exploit a rich set of distributed service components that can be composed to provide each user with a multitude of application level services. In opportunistic networks, tasks such as content sharing and service execution among remote devices are facilitated by intermediate devices capable of short–range wireless connectivity, also called *relays* that receive, move around, and forward the data.

To enable effective collaboration in such an environment, we make three novel contributions: (i) an *adaptive forwarding scheme*, ProxiMol, to select suitable relays for data transfer; (ii) a *distributed mechanism* to select services that can be composed; and (iii) an *incentive–compatible credit scheme*, CRISP, to promote participation of relays to forward data. These contributions resolve key challenges in opportunistic networks as in existing works, data transfer is not adaptive to changing user behavior, service composition is not performed on remote devices in the absence of a connected path, and rewards are not incentive–compatible (i.e. rewards do not promote honest behavior of the users).

ProxiMol is a novel forwarding scheme leveraging two simple facets of users in opportunistic networks – some users have better likelihood of message delivery due to higher *mobility*, while others do due to their locations *proximity to destination*. Key contributions to the design of ProxiMol include: (i) a model to infer users location over time from diffusion (a measure of mobility); (ii) an analytical result to estimate distance between users; and (iii) an empirical method to estimate diffusion of a user. ProxiMol improves delivery ratios (10–20%) and reduces delays by up to 50%, when compared against previously proposed algorithms.

The proposed service composition algorithm derives efficiency and effectiveness by taking into account the *estimated load* at service providers and *expected time* to opportunistically route information between devices. Based on this information the algorithm decides the best composition to obtain a required service. It is shown that using only local knowledge collected in a distributed manner, performance close to a real–time centralized system can be achieved. Scope and applicability of the service composition algorithm in a range of mobility characteristics are established through extensive simulations on real/synthetic mobility traces.

CRISP is the first credit scheme in which both routing and forwarding are designed to be incentive compatible in opportunistic networks, i.e., honest behavior of users maximizes their profit. Data transfer and loss in opportunistic networks are modeled as a linear generalized flow network where flow maximization leads to optimal relay behavior. This optimal behavior is made incentive compatible by requiring a relay to make a specific payment upon receiving the data and earn reward for forwarding the data. Simulations on real and synthetic mobility traces validate our analysis, showing a significant gain in throughput when compared with the existing credit schemes that are not incentive compatible.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

In this dissertation, we provide our research on design and analysis of data forwarding, service composition and incentives for collaboration in opportunistic networks. In this chapter, we introduce the opportunistic networks and the motivations for our research. Then, we overview the key research issues in opportunistic networks followed by the contributions of this dissertation. The introduction is concluded by an outline of rest of the dissertation.

1.1 Opportunistic Networks

An opportunistic network is formed by contacts among devices carried by people in different environments, e.g., a social event or gathering (art exhibition, state fair or local festival), a park (walking / cycling trails), outdoors (streets in a city, picnic resorts, university campus) etc. The primary challenge in opportunistic networks is to transport data packets in a reliable and efficient way despite the absence of a connected path between the end points. Data transmission is critical to every fundamental service that an opportunistic network can provide, including content sharing, dissemination, collection, and collaboration [18, 29, 43, 67]. The underlying mechanism used to forward data in these networks is to rely on any device in the area with short–range wireless connectivity to receive the data, expect the movements of the devices to make new connections possible, and to use these new connections to forward data onto other devices with the goal of eventually finding a path to the destination.

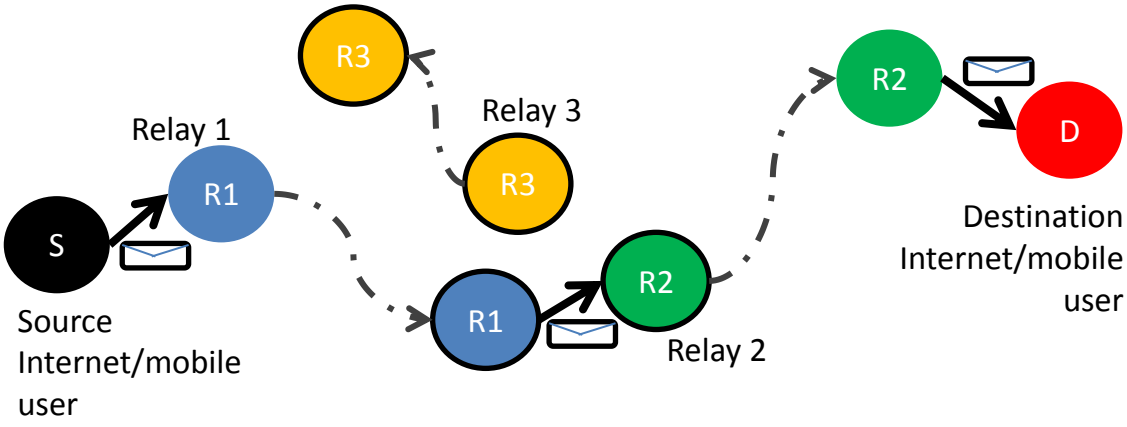

Figure 1.1 Data forwarding in Opporutnistic Networks

Figure 1.1 shows how data is forwarded from a source to a destination in the absense of an end-to-end connected path. The source transmits data to relay 1 which moves to another location to forward the packet to relay 2 which moves further along the path to eventually deliver the data to destination. If relay 2 forwards the data to only relay 3, then the data may not be delivered to the destination. Therefore, it is important to carefully select relays that can forward the data.

1.1.1 Why Opportunistic Networks?

People need useful information and services, which may be acquired *locally* or on the world wide web. With the increasing number of devices, and limited bandwidth available for 3G and 4G connections, researchers in academia and industry are focusing on mechanisms that can offload some of the load to the wifi network [9]. Opportunistic networks take it a step further such that data from 3G network is offloaded to a Wi–Fi network which is then delivered to a device using opportunistic contacts [41]. Opportunistic contacts can also be leveraged in scenarios when information is not available on the Internet (e.g., deals and services offered by stores and

people in the physical proximity of a device [67]) or connection to the Internet itself is not possible (e.g., when wireless service is unavailable at some location).

1.1.2 Practical Implementations of Opportunistic Networks

Opportunistic contacts between mobile users enable a mechanism to provide local information and services, both in the presence of traditional wireless infrastructure and in its absence. With increasing number of handheld devices capable of short-range wireless connectivity, opportunistic networks have received a great deal of attention from the research community. Several research projects, such as Situated and Autonomic Communications (Haggle) [4], Consumer Generated Mobile Wireless Media (Crowd) [2], Delay Tolerant Networking Research Group (DTNRG) [3] and others focus on opportunistic networking paradigm as it provides avenues for content sharing, dissemination and collaboration in open environments where citizens interact.

A detailed overview of using bluetooth communication for opportunistic networks is provided in [71]. Another possibility is to use traditional Wi–Fi or Wi–Fi–Direct[1](a new standard for ad hoc communication). In addition, collaboration between proximal devices can be facilitated by future technologies (like FlashLinq[2]) that enable continuous, autonomous, and energy-efficient discovery of devices and services in the environment (distances up to a few hundred meters). FlashLinq is currently being developed at Qualcomm, and a prototype chip is being tested with SK Telecom in Korea.

[1]http://www.wi-fi.org/discover-and-learn/wi-fi-direct

[2]http://www.qualcomm.com/media/videos/flashlinq

AllJoyn[3] is another peer-to-peer technology (software solution) that enables ad hoc, proximity-based, device-to-device communication without the use of an intermediary server. It aims to provide simple device and service discovery, managed networking and message routing with applications in multi-player gaming, social media sharing and multi-user productivity tools.

1.2 Research Issues

To collaborate effectively in an opportunistic network, a user needs to be able to send/receive data from devices in the environment even in the absence of an end–to–end connected path. Since there may not be a direct connection to exchange information in real–time, a user also needs a mechanism to select suitable devices providing the required services. If no single (nearby) device is providing the required service, there may exist a combination of service components on several devices that meet the requirements of the requested service (e.g. to meet the requirement of compression and encryption, one device compresses the file and another device encrypts the compressed file). This concept is formally known as service composition (i.e. combining/aggregating services). Performing service composition (which essentially requires transferring data between devices) is only possible through cooperation of other devices in the environment which provide resources (bandwidth, memory, computation time etc.) to *relay* data. Thus, to summarize, effective collaboration requires an efficient mechanism to transfer data, compose services and promote user participation. Next, we discuss the key challanges in data forwarding, composition of service components that are distributed on different devices in the network, and provision of proper incentives for users to take part in data forwarding.

[3]http://www.qualcomm.com/media/videos/alljoyn-overview-video , https://www.alljoyn.org/

1.2.1 Data Forwarding

Opportunistic networks face the challenge of intermittent connectivity. In the absence of end-to-end paths, messages are received, stored and forwarded by zero or more intermediate nodes until the destination node is reached. For efficient forwarding, it is desirable to select intermediate nodes, referred to as relays, that have a better likelihood of delivery. This dissertation provides formal basis and rigorous analysis to fill a gap in literature by defining the key tradeoff between *destination proximity* and *node mobility* for enabling efficient communication in open, dynamic environments.

In scenarios where nodes are humans carrying a device, mobility plays a fundamental role in making routing decisions [23]. As generally there is some correlation of a node's past movement profile with its future encounters, there have been attempts to model such basic properties as inter-contact times [45], rates of encounters [64] (and others) which can be used to make intelligent routing decisions. However, it is difficult to accurately model these properties as there is still no large scale deployment of nodes to collect movement traces. Also, the behavior of a node itself can change at different locations and times [45] making it harder to generalize any one model. As a result, a number of mobility models have been proposed that focus on one or other aspects of such networks [36, 57, 58, 73]. Consequently, a large number of routing and forwarding schemes have emerged with certain design goals. These schemes can be classified based on the delay that can be tolerated [31, 46], type of participating nodes [14], mobility characteristics of nodes [59, 64, 83] and level of connectivity available in the network [42, 75]. The inadequacy of assuming homogeneous mobility characteristics in realistic environments has recently been pointed out in [80], however, its proposed metrics in turn rely on distinct classes of nodes in a highly heterogeneous setting. While realistic environments are not entirely homogeneous, they

are not highly heterogeneous (with distinct classes of nodes) either. It is important to adapt to environment conditions (level of connectivity, mobility characteristics of nodes) that can vary over time and location.

1.2.1.1 Research Problem:

In order to reach users in open environments, it is critical to define *how to identify a relay with better likelihood of delivery* based on time of last contact [34, 39], temporal distance [82, 83], transitive heuristics [14, 59], rate of encounter [64] or sociability metrics [80] etc. and still utilize partial paths [42, 75]. The forwarding scheme, ProxiMol, proposed in this dissertation solves precisely this problem. Our analytical results show that [34, 39, 59, 82, 83] work well in homogenous environments as they roughly approximate distance from destination while [64, 80] are efficient in highly heterogeneous settings as they estimate diffusion (a measure of mobility) of a node in order to compute likelihood of delivery.

1.2.2 Composition of Distributed Services

In recent years, the number of multi-functional, personal smart devices has been increasing at a high rate. The possibility of such devices coming within communication range of each other is enhanced by the presence of multiple embedded radios. While such opportunistic contacts between pairs of devices have been exploited by the opportunistic networking paradigm [64, 70, 83], exploiting resources to execute remote services is an area of research that has received attention in recent times [68, 69].

Heterogeneous resources available on devices can be abstracted as services to simplify the interface and have platform independence. Services from multiple devices can be composed to provide enhanced functionality [29, 48]. For example, to complete a task like '*take a picture* and share it on a *map* in my *current location*', services

offered by a camera, GPS receiver and map provider need to be composed [22]. Service composition will be applicable in several areas - pervasive healthcare, intelligent transportation systems, crisis management, etc., [29].

1.2.2.1 Research Problem:

The creation of a services-rich environment by effectively making services available on each device, accessible to applications on other devices in opportunistic environments is a major challenge due to intermittent connectivity. This dissertation investigates efficient service composition in opportunistic networks where services are likely to be available on user devices in physical proximity of a user. Such environments include parks, malls, streets in a city or other social gathering places where citizens interact. In particular, an efficient mechanism for selecting services to complete a request and forward service parameters to corresponding devices (that provide selected services) in an opportunistic network is proposed.

1.2.3 Incentives for Participation

Numerous studies have investigated opportunistic routing in fully cooperative settings, in which each node participates in data forwarding to the best of its ability. Careful planning based on nodes' past and estimated future connection and movement behaviors can aid in efficient forwarding in order to extract the most benefit out of each node's available resources [46, 59, 83]. In many realistic settings, however, we must anticipate selfish users who will not freely receive and send data at the cost of their devices' battery power and memory without some benefit to themselves. Therefore, it is necessary to have incentive mechanisms to motivate users. Additionally, malicious users who seek to undermine the network's capabilities through manipulation of the

routing system, e.g., by falsely claiming short routes to all destinations and dropping all data in a black hole attack or by simply flooding the network, must be discouraged.

Researchers have proposed a range of schemes to address these issues in opportunistic networks [55, 66, 94] as well as other network settings, such as ad hoc networks [87, 88, 92, 93]. The most promising approach to address these issues is to provide *credit* to nodes that participate in the forwarding process and allow only nodes with credit to generate and send their own data packets[4]. Existing schemes in opportunistic networks, however, are not incentive compatible in routing and are vulnerable against a small group of colluding attacker nodes [21, 55, 56, 60, 66, 72, 94]. Also, existing schemes that are incentive compatible, are designed to work in ad hoc networks in which the routing path is known [87, 92] or in mesh networks in which the routing and forwarding mechanisms are considered separately [27, 89].

1.2.3.1 Research Problem:

To date, no incentive–compatible scheme has been designed to operate in opportunistic environments in a fully distributed way without: i) an online trusted authority and ii) *a priori* knowledge of routes. Clearly, it is a challenge to have an online trusted authority and acquire prior knowledge of routes in opportunistic networks.

1.3 Contributions

To address the research questions outlined above, this dissertation makes pioneering contributions in the area of data frowarding, service composition and provision of incentives that are discussed next.

[4]we use the term data, packets, and data packets interchangeably.

1.3.1 Proximity and Mobility Estimation for Efficient Forwarding

A series of opportunistic contacts in space and time among devices carried by mobile users can be utilized to forward messages from one user to another, in the absence of an end-to-end connected path. Existing routing metrics exhibit efficient performance in either homogeneous (users have similar mobility characteristics) or specific heterogeneous (varied mobility characteristics) scenarios. However, in practice, behavior of users changes at different locations and times, making it hard to generalize any one routing algorithm.

The adaptive forwarding scheme, ProxiMol, proposed in this dissertation leverages the fact that while some users have better likelihood of message delivery due to *higher mobility*, others do due to their location's *proximity to destination*. Key contributions of ProxiMol include:

1. a model to infer user's location over time from its diffusion (a measure of mobility),
2. an analytical result to estimate distance between users, and
3. an empirical method to estimate diffusion of a user.

These are used to compute the likelihood of delivery taking into account both the mobility of a user and her proximity to destination. In addition to this robust forwarding scheme, a novel concept of *disconnected distance* is introduced that captures partial paths in networks with moderate level of connectivity. ProxiMol improves delivery ratios (10-20%) and reduces delays by up to 50%, when compared against previously proposed algorithms, in user environments that range from relatively homogeneous to highly heterogeneous settings.

1.3.2 Service Composition in Opportunistic Networks

Pervasive networks formed by users' mobile devices have the potential to exploit a rich set of distributed service components that can be composed to provide each user with a multitude of application level services. However, mobile and pervasive networks suffer from intermittent connectivity, disconnections and partitions, such that opportunistic networking techniques are required to enable communication. This poses novel challenges to service composition techniques.

While several works have discussed middleware and architectures for service composition in well-connected wired networks and in stable MANET environments, the underlying mechanism for selecting and forwarding service requests in the significantly challenging networking environment of opportunistic networks has not been addressed. The solution comprises three stages:

1. selecting an appropriate service sequence set out of available services to obtain the required application level service;
2. routing results of a previous stage in the composition to the next one through a multi-hop opportunistic path; and
3. routing final service outcomes back to the requester.

The proposed algorithm derives efficiency and effectiveness by taking into account the estimated load at service providers and expected time to opportunistically route information between devices. Based on this information the algorithm decides the best composition to obtain a required service. It is shown that using only local knowledge collected in a distributed manner, performance close to a real-time centralized system can be achieved. Scope, performance guarantee, and applicability of the service composition algorithm in a range of mobility characteristics are established through extensive simulations on real/synthetic traces.

1.3.3 Collusion-Resistant Incentive-Compatible Forwarding

In opportunistic environments, tasks such as content sharing and service execution among remote devices are facilitated by relays (devices with short–range wireless connectivity) that receive data, move around, and then forward the data. To achieve high throughput, it is important to secure forwarding and provide incentives for participation by relays. However, it is extremely challenging to monitor the behavior of relays in an opportunistic network due to sparse connectivity. Existing schemes do not work when selfish/malicious relays collude with each other to forge routing metrics, drop useful data, flood the network, and/or attempt to earn extra reward.

The credit scheme, CRISP, presented in this dissertation is the first in which both routing and forwarding are designed to be incentive compatible. To design the scheme, the data transfer and loss in an opportunistic network are modeled as a non–linear generalized flow network. Then, optimality conditions for flow maximization describe the optimal behavior of a relay. This optimal behavior for forwarding is made incentive compatible by requiring a relay to make a specific payment upon receiving the data and earn reward on forwarding the data. To make the scheme collusion–resistant, the reward for forwarding a packet is based on the routing utilities of relay nodes instead of a fixed amount and digital signatures are used to enforce specific conditions on routing utilities of relay nodes. Finally, a distributed payment framework (DPF) is proposed to implement CRISP in a completely distributed and opportunistic environment. DPF is the first framework that does not require a trusted third party to maintain credits. DPF maintains consistency by using vector clocks and is secured by using Merkel trees. Simulations on real and synthetic mobility traces validate our analysis, showing a significant gain in throughput when compared with the existing credit schemes that are not incentive compatible.

1.4 Dissertation Outline

The remainder of dissertation is organized as follows. We start with the review of related literature in Chapter 2. We present our solution for efficient data forwarding in Chapter 3. Using key insights in the previous chapter, we present the design and analysis of distributed algorithm for service composition in Chapter 4. Research on provision of incentives that promote honest behavior of users to participate in the data forwarding is discussed in Chapter 5. Conclusion and directions for future research are provided in Chapter 6.

CHAPTER 2

RELATED LITERATURE

In this chapter we survey the literature related to this thesis. The survey is organized under three categories: (1) Data forwarding, (2) Composition of distributed services, and (3) Incentives for participation.

2.1 Data Forwarding

2.1.1 Homogeneous Mobility Characteristics

Simplest forwarding scheme is to wait until a direct contact with destination occurs. Likelihood of successful delivery can be increased by spreading a certain number of copies in the network. This concept is presented in Spray and Wait [81]. Another criteria to estimate likelihood of delivery is to use time since last encounter with the destination [34]. It has been shown to provide good location approximation in dense mobile networks [39]. The caveat of this approach is the slow warm-up time (i.e. it takes a long period of time before every node has at-least one contact with every other to make an estimate of its location). To reduce warm up time, [83] presented a timer transitivity (nodes exchange timer values for all destinations) that works better than probabilistic heuristics [59]. This idea is further used in [82]. However, sole use of this criteria can still be harmful in some scenarios as recently pointed out in [20], where nodes with minimum last encounter time are least likely to go back.

2.1.2 Heterogeneous Mobility Characteristics

On the other side of the spectrum, the performance of simple Spray and Wait [81] can be improved by carefully selecting the nodes on which these copies reside (more social first). This improves performance by 4-5 times in highly heterogeneous environments (nodes with varied mobility characteristics) [80]. A similar idea is presented in [64] where it is shown that simple use of encounter rate can achieve high improvements in networks where a subset of nodes is highly mobile. While most of these metrics implicitly assume sparse network connectivity, recently it has been shown that partial paths can be leveraged to improve forwarding efficiencies by transmitting messages to the point of disconnection between temporarily isolated node clouds [42] in less sparse networks.

2.1.3 Social Network Analysis

There is also much work on forwarding metrics using social network analysis [31, 46] which performs efficiently only over large delays (order of hours to days) when participating nodes have social interaction. Similarly there are history based prediction schemes that capture this same effect by using the strong correlation of nodes' future trajectory with locations previously visited [15, 90]. These are not applicable to our analysis on open environments where participating nodes may not have any social interaction history on previous days and can potentially be visiting some place for the first time. However, these schemes can augment the performance of ProxiMol when such context is available and large delays can be tolerated.

2.2 Composition of Distributed Services

2.2.1 Single Service Provisioning

When two devices are within communication range, there exists an opportunity to exploit each other's services. If the contact period is sufficiently long, nodes can utilize each other's resources to execute remote services. On the other hand, if the contact time is insufficient, service parameters can be exchanged during the contact, service can be executed offline and service outputs transmitted to the requester using opportunistic contacts. In Passarella et al. [69], multiple copies of a request are initiated to minimize delay in service execution and to increase robustness but only one-hop communication (i.e. a requester waits until a direct contact with service provider) is used. Also, a few other works propose efficient schemes and use multi-hop paths for service discovery and invocation in opportunistic networks by using a set of proxies [52] or location of the user [79]. However, composition of multiple services is not considered in these works.

2.2.2 Composition of Multiple Services

For composition of services, a number of middleware frameworks and architectures are proposed but these do not discuss the actual forwarding mechanism [22, 32, 48]. Routing is considered in [33, 86] by modeling the expected colocation time between nodes based on past interactions. However, service composition fails if participants do not remain directly connected. In contrast in this thesis, service composition is performed using multi-hop paths that can relay information to the service providers and back to the service requestor even when an end-to-end connected path does not exist. Furthermore, alternative paths are considered to increase the success of composition.

2.2.2.1 Semantics and Ontologies

There has been a significant research on semantics and ontologies to describe services and compositions (see [24] for survey and issues of service composition in mobile environments). Services to be composed can either be strictly defined in the request (static composition) [40], or can be computed at run-time (dynamic composition) based on the request [48]. Dynamic composition looks for alternate sets of services in the current environment that can be composed to provide the required functionality. Services are modeled as directed attributed graphs [48], to find possible compositions. We leverage this work for service representation to find possible compositions and explore mechanism to execute such compositions in an opportunistic network.

Passarella et al., [68, 69] investigated optimal policy for service execution in opportunistic networks. An optimal policy is derived for the level of replication to receive service results in minimal time but only a single service is requested and the service requester waits until a direct contact with service provider. In contrast, the algorithm presented in this thesis uses a single instance of service request to compose multiple services, collects all information in a distributed manner, and uses multi-hop forwarding paths to upload requests to and download results from service provider. Since, multihop forwarding paths are used, next we describe the existing schemes to forward data in opportunistic networks that are applicable to our analysis.

2.2.2.2 Forwarding Schemes

Several schemes for forwarding messages in opportunistic networks have been proposed in the literature. Two key metrics used in open environments where participating nodes may not have social interaction history include, *time since last encounter* with destination [34] and *rate of encounter* [64, 80] . Time since last encounter provides good location approximation in dense mobile networks [39] but has a slow warm up time (i.e. it takes a long period of time before every node has made at least one contact with every other to make an estimate of its location). To reduce warm up time, Spyropoulos et al. [83] presented a timer transitivity (nodes exchange timer values for all destinations). Whereas, use of encounter rate provides good delivery rates in heterogeneous environments by identifying nodes that are highly mobile.

2.3 Incentives for Participation

2.3.1 Incentive Schemes in Adhoc Networks

In ad hoc networks, the earliest work proposes use of a virtual currency called 'nuglets' that are added to the generated packet and each relay on the forwarding path takes its share [17]. However, this scheme requires tamper–proof hardware. An elegant solution is proposed in [91] that provides credit to relays that forward a packet. The payment for the source and the relays is adjusted so that following the forwarding protocol is in their best interest. This solution, however, does not result in the optimal selection of the routing path. Therefore, a technique based on VCG mechanism is proposed to select optimal routing paths as being truthful maximizes a nodes reward [6, 92]. [87] also considers that the source has limited budget and provides a mechanism that makes routing/forwarding a Nash equilibrium strategy.

The earliest work proposes use of a virtual currency called 'nuglets' that are added to the generated packet and each relay on the forwarding path takes its share [17]. This requires the estimate of number of relays in path at the source, which may not always be available. To alleviate this problem, an improved technique is to use counters at each relay that are incremented upon each forwarding [17]. However, this scheme requires tamper-proof hardware and is no longer deficit-free (credit can be generated/destroyed in the network). An elegant solution is proposed in [91] that provides credit to relays that forward a packet. The payment for the source and the relays is adjusted so that following the forwarding protocol is in their best interest. This solution, however, does not result in the optimal selection of the routing path, i.e. a node can change its routing metric to receive more packets and then forward them to earn more reward. Therefore, a technique based on VCG mechanism is proposed to select optimal routing paths as being truthful maximizes a nodes reward [6]. A more detailed and corrected analysis is provided in [92] that treats routing and forwarding as a two-stage game. It makes 'following the routing protocol' to be a dominant strategy and 'forwarding' to be a Nash equilibrium strategy of relays. It also proves that no protocol can make 'forwarding' the dominant strategy (DS) due to the in-distinguishability of failure at sending or receiving side of pairwise communication. However, works in [6,92] only focus on the game from the perspective of a relay. [87] considers the source with limited budget and relays with fixed cost of transmission, and provides a mechanism that makes routing/forwarding a Nash equilibrium strategy. The condition on routing is relaxed from DS, as DS requires a lot of extra payment to relays.

2.3.2 Incentive Schemes in Opportunistic Networks

For opportunistic networks, it is shown in [13] that replication provides a good defense against black-hole and flooding attacks. On the contrary, [28] shows that in combination these attacks can degrade the network performance a lot. Credit based schemes limit flooding as they require the source to pay in order to generate messages. Also it is easier to monitor dropped packets in a connected environment like ad-hoc networks, as compared to opportunistic networks. Therefore, several techniques have been proposed to address black-hole attack in an opportunistic network. [66] requires the relay to pay for receiving a message, whereas it gets its reward by requesting a higher payment from the next relay. Finally, destination pays for all the reward, and a credit authority decrements source, and increments destination's credit account. The exact amounts of payments are not discussed and the scheme requires a credit authority to be online only in order to resolve payment conflicts. [60, 94] introduce the concept of layered coins for payments such that the credit authority never needs to be online at the time of transaction. [21] secures information of each encounter in an un-forgeable ticket that is signed by the private keys of encountering nodes. These tickets are then used to estimate delivery likelihood of nodes in [55]. While [55] secures the routing metric, a relay can still drop all the messages it receives using its true routing metric, and there is no check on that. [72] addresses this limitation in [55], by recording the information about exchanged messages in encounter tickets. Whereas [56] addresses the same limitation by using ACK messages and a more direct reputation-assisted forwarding mechanism.

2.3.3 Collusion of Multiple Nodes

2.3.3.1 Collusion in Adhoc Networks

Works in [6, 87, 92] still assume that nodes do not collude to increase their reward. [93] provides a mechanism to defend against collusion when colluding nodes do not completely trust each other and [88] shows that no mechanism can defend against collusion when colluding nodes can share each other's profit. Owing to this negative result on defense against collusion, [7] proposes a framework that monitors the network for special cases of collusion. However, such a framework cannot be used in opportunistic networks because observing network traffic is not possible due to disconnections and sparse connectivity. All these works show that is extremely challenging to provide defense against collusion of multiple nodes in ad hoc networks - even more so in an opportunistic network with disconnections and sparse connectivity.

2.3.3.2 Collusion in Opportunistic Networks

All these techniques [21,55,56,60,66,72,94] fail when multiple nodes can collude with each other. In collusion, it can be assumed that nodes will share their private keys and earned rewards with each other. Therefore, some relays can add other relays in the path even when others do not take part in actual forwarding [21,55,56,60,66,72, 94]. This way, nodes can earn additional credit and use it to launch flooding or black-hole attacks. Similarly, colluding nodes can easily forge encounter tickets [21, 55], packet exchange records [72] or reputation values [56] by using their private keys and signing off fake receipts.

CHAPTER 3

PROXIMITY AND MOBILITY ESTIMATION FOR EFFICIENT FORWARDING

In this chapter, we present our reserach on efficient data forwarding in opportunistic networks. An introduction to opportunistic networks and background on current literature has been provided in Chapter 1 and 2 respectively.

3.1 Introduction

We propose an adaptive forwarding scheme called ProxiMol (Proximity and Mobility Estimation). To this end, ProxiMol enables efficient communication by:

1. defining the tradeoff between proximity of location and mobility of a node (measured by its diffusion) to compute delivery likelihood;
2. adapting to changing environment conditions and user behavior based on design; and
3. utilizing partial paths [42, 75] in moderately connected networks by estimating disconnected distance.

The first part of the analysis uses a node's diffusion to infer its future location [Section 3.3.1]. Note that computing diffusion does not require repeated contact patterns or recurrent location landmarks. The likelihood of delivery is computed by using an estimate of a node's diffusion and its distance from the destination [Section 3.3.2]. An analytical result is derived to estimate distance between nodes and an empirical method is proposed to locally measure diffusion of a node [Section 3.3.3]. These results are then used to develop a simple forwarding algorithm, ProxiMol, in Section 3.4.

Our approach for deriving ProxiMol is unique in two key aspects:

1. *ProxiMol is derived from the mobility model itself*;
2. *Power-law distributions of flight lengths, pause durations between flights and inter-contact times of the nodes are used.*

The key aspect (1) is important because it is used to define relationship between mobility characteristics that play a key role in efficient forwarding; whereas (2) has only been highlighted in recent works [19,73]. Most existing works assume exponential inter-contact times, while none of the routing metrics take into account the effect of power-law flight-lengths and pause-time durations that have been shown to be statistically significant in [73].

To validate our analytical findings, simulations are carried out on synthetic as well as real mobility traces from State Fair, Disney World Orlando, and KAIST. These traces satisfy conditions of varied mobility characteristics and different levels of connectivity. ProxiMol shows increase in message delivery ratios by 10-20% and decrease in the average delays by up to 50% when compared against previously proposed algorithms. ProxiMol complements existing routing protocol frameworks [54] and provides better estimation to optimize tradeoff between available network resources [8]. It can also be used to improve predictability and estimates of contact quality for adaptive schemes [63,76] in more heterogeneous environments with power-law inter-contact times.

3.2 System Description

3.2.1 Mobility Model

It has been shown that real mobility traces follow a power law distribution of Inter-Contact Times (ICTs) in contrast to previously assumed exponential distribu-

tion [23]. Since random way-point and random direction models lead to exponential ICTs [77], they are not suitable to represent user mobility. In addition, it has been shown that real mobility traces have power-law flight-lengths and pause-time distributions [73]. In [53], a detailed study compares a large number of existing mobility models. The study reveals that only Truncated Levy Walk (TLW) [73] and SLAW [53] show all of the above mobility characteristics. We use TLW as it captures the statistical properties of real mobility traces and still remains analytically tractable. In [73] Rhee et al., have shown that truncated levy walks with trapping (pause duration between successive steps) can describe human mobility, and have power-law distribution of ICTs, flight-lengths and pause times. This model describes the movement of a node at constant speed as a sequence of independent steps with some length h at an angle θ followed by a pause of duration z such that

$h \in (0, \tau_h)$ under distribution $\Lambda_h(h) \sim 1/h^{1+\alpha}$,

$\theta \in [0, 2\pi]$ under uniform distribution, and

$z \in (0, \tau_z)$ under distribution $\Lambda_z(z) \sim 1/z^{1+\beta}$,

where $\alpha, \beta \in (0, 2)$ and the truncation points τ_h and τ_z are the maximum length and pause durations respectively. Note also, that above distribution only shows asymptotic behavior. The characteristic function of the actual distribution with scale factor o and exponent ζ is given by,

$$\varphi(t) = e^{-|ot|^{\zeta}} \tag{3.1}$$

where ζ corresponds to exponents α and β in asymptotic behavior of step length and pause time distributions. For the special case of $\zeta = 2$, equation (3.1) reduces to normal distribution with variance $2c$ but otherwise (3.1) has infinite variance without truncation points.

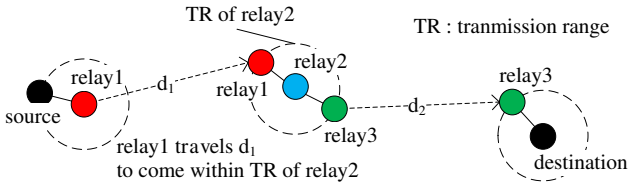

Figure 3.1 Disconnected distance between source and destination = $d_1 + d_2$

3.2.2 Forwarding Path

The key problem in ProxiMol is, how to decide a candidate relay, with better likelihood of delivery for a particular destination. There can be a number of possible paths, between a pair of nodes, using different relays. Each relay in the path buffers the message and moves around in the network until it encounters a "better" relay or the destination itself. In order to make a contact, two nodes need to be within each other's transmission range. Thus in a forwarding path, between two contact opportunities, a message must *travel* some distance towards the destination. Here we refer to *travel* as the distance traveled by a message while it resides in the buffers of a series of mobile relays. As transmission delay of messages between connected nodes is negligible in comparison to total delays experienced by messages in typical opportunistic networks, the primary contributor of delay is the time taken to traverse this disconnected distance between a source and destination. Fig. 3.1 shows the disconnected distance (distance in disconnected space) where *relay1* and *relay3 travel* some distance to complete a forwarding path from source to destination.

As nodes can have different mobility characteristics - velocities, step length distributions etc., they move around different amounts of distance in a given time. Thus, an optimal forwarding path is the one that takes least amount of time for the message to travel from source to destination when such a path exists. Knowledge of future contacts can be used to compute shortest path. But when such knowledge is not available, expected results for rest of the networks can be used to find an efficient

forwarding path on a per-hop basis in comparison with source-routing [47]. Therefore, our aim is to minimize the time taken for the message to *travel* disconnected distance between source and destination by selecting suitable (more mobile and closer to destination) relays at each step.

3.3 Relay Selection

Using the model described in the previous section, we derive results that approximate a node's future location relative to its current position. It follows that location depends on elapsed time and node's mobility (measured by its diffusion). Together with location approximations and an estimate of nodes' disconnected distance from the destination, it is possible to compare two nodes for their likelihood to deliver a message to some destination. In the following sub-sections, analytical results validating these observations are presented.

3.3.1 Node's Location Approximation

Ideally, it is desirable to know the location of a node as a function of time when it is following the mobility pattern described in Section 3.2.1. More precisely, we require $\phi(\vec{r}, t)$ which is the probability density of a node's location $\vec{r} \in \Re^2$ at time t when it starts from origin at $t = 0$. Using the Continuous Time Random Walk Formalism [44], in [50] Kim et al. have shown the complex structure of $\hat{\tilde{\phi}}(\vec{w}, s)$ (Laplace-Fourier Transform of $\phi(\vec{r}, t)$ used for simplifying convolution of step distributions) that defies any attempt to solve for exact equation. Thus the option to approximate $\phi(\vec{r}, t)$ is justified.

To approximate $\phi(\vec{r}, t)$ as a normally distributed function is simple and follows from the Central Limit Theorem (CLT):

- Each node selects step lengths from the same distribution for itself; hence, its position after a few steps is a sum of *iid* step lengths.
- Truncation points exist for the heavy tail step-length and pause-time distribution which makes their mean and variance finite.

Thus CLT is applicable.

Even though the position of a node after few steps is normally distributed around starting location, to relate it with time is more complex as it requires cognizance of the effect of pause durations after each step (trappings). Since number of steps taken directly scales the variance of normal distribution, we study the variance of $\phi(\vec{r}, t)$ and its temporal behavior.

The variance is sometimes also referred to as the mean square displacement in the literature. In [50] Kim et al. theoretically show the asymptotic behavior of mean square displacement as a function of time $M(t)$ for different ranges of the value of exponents in step-length and pause-time distributions to be given by,

$$M(t) \sim t^{\gamma} \tag{3.2}$$

where the value of exponent γ depends on α, β and is provided in Table 3.1 for the range which is relevant to actual values of these exponents as measured from traces. These relations are corrected for the index shift due to different notations used in [50].

Table 3.1 Value of γ for different ranges of α, β

	$1 < \alpha < 2$	$\alpha > 2$
$0 < \beta < 1, \beta < \alpha$	$2 + \beta - \alpha$	β
$1 < \beta < 2$	2	$3 - \alpha$

Values of α and β have been measured in [73] using real mobility traces from five sites. These include two university campuses (NCSU in US, KAIST in Asia), New York City, Disney World and State Fair. They also measure the exponent of mean square displacement and found that its value decreases after about thirty minutes. These values are labeled as γ_{init} and γ_{mid} in Table 3.2. In addition, values of α and β are measured in [50] using AP-based traces from the University of California at San Diego (UCSD) [62] and Dartmouth College [1]. Using the relationship from Table 3.1, the value of γ is computed for first five sites and α for last two sites. All these values are summarized in Table 3.2 (theoretically computed values are bold faced).

Table 3.2 Values of α, β, γ from mobility traces

	α	β	γ	γ_{init}	γ_{mid}	$\alpha - \beta$
NCSU	1.77	0.99	**1.22**	1.32	0.62	**0.78**
KAIST	2.08	0.45	**0.45**	1.30	0.74	**1.63**
NYC	1.75	0.49	**0.74**	1.52	0.73	**1.26**
Disney	2.20	0.80	**0.80**	1.23	0.82	**1.40**
Fair	1.86	1.68	**2**	1.08	0.19	**0.18**
UCSD	**1.73**	0.38	0.65	-	-	**1.35**
Dart.	**1.54**	0.23	0.69	-	-	**1.31**

As seen from Table 3.2, the measured values of these exponents typically lie in the range $\alpha \in (1, 2), \beta \in (0, 1)$ and $\alpha - \beta \approx 1$. Then using theoretical relationship from Table 3.1, the value of exponent γ is given by,

$$\gamma = 2 + \beta - \alpha \approx 1 \tag{3.3}$$

Also, as seen from the measured values $\gamma \in (0.7, 1.3)$ and the fact that nodes show slightly super-diffusive ($\gamma_{init} > 1$) behavior for a few minutes after which they become

sub-diffusive ($\gamma_{mid} < 1$); $\gamma \approx 1$ stands as a good general approximation to node's diffusive behavior and can be used to approximate their location over time.

Based on CLT, theoretical analysis of mean square displacement as a function of time, and empirical evaluation, we approximate $\phi(\vec{r}, t)$ as normally distributed with variance linearly increasing with time. Hence, at time scale of a few steps, $\phi(\vec{r}, t)$ is given by,

$$\begin{aligned} \phi(\vec{r}, t) &= \frac{|\vec{r}|}{2\pi Dt} e^{-\frac{|\vec{r}|^2}{2Dt}} \\ \Rightarrow \phi(r, t) &= \frac{r}{Dt} e^{-\frac{r^2}{2Dt}} \end{aligned} \tag{3.4}$$

where $\phi(r, t)$ is the probability density of a node's location at the distance r from the origin at *any* angle θ and corresponding to (3.2) and (3.3) the variance is given by,

$$M(t) = \int_0^\infty (r^2 - r)\phi(r, t)dr = ((4 - \pi)/2))\, Dt \tag{3.5}$$

where D is the diffusion constant that depends on a node's mobility (distribution of step length, pause time and velocity of the node). Equation (3.4) is a Rayleigh distribution which in this case is same as bivariate normal distribution when using Cartesian coordinates,

$$\phi_{X,Y}(x, y, t) = \frac{1}{\sqrt{2\pi Dt}} e^{-\frac{x^2}{2Dt}} \frac{1}{\sqrt{2\pi Dt}} e^{-\frac{y^2}{2Dt}}.$$

Polar form is used here as it gives a closed form integral for derivations in the next section. Note that the above equation describes Brownian motion in two dimensions. This can be explained by the fact that even though movement of real nodes is more diffusive ($\alpha < 2, \gamma > 1$) as compared to Brownian motion ($\alpha = 2, \gamma = 1$), pause durations after each step make it less diffusive ($\alpha < 2, \beta > 0, \gamma < 1$) on the scale of few steps. Therefore, the net displacement after some time can still be approximated by Brownian motion.

3.3.2 Delivery Likelihood

Approximation for a node's displacement from its current position can be used to measure its likelihood to deliver a message to a particular destination. If a comparison between two nodes (a, b) can be made, it can then simply be extended to compare any number. Let the two potential nodes be at distances d_a and d_b respectively from the destination. Also let the next potential relay in forwarding path to destination be at distance d_r from the destination such that it lies somewhere between the nodes (a, b) and destination. Since nodes can have different mobility characteristics, their corresponding diffusion constants are D_a and D_b, respectively.

To compute which node is more likely to reach next relay first, we compute each node's probability to travel at least distance $(d_a - d_r)$ and $(d_b - d_r)$ respectively in time t. Then *node a* has better likelihood when

$$\begin{aligned} \frac{P[r(t) > (d_a - d_r)]}{P[r(t) > (d_b - d_r)]} &> 1 \\ \Rightarrow \frac{\int_{d_a - d_r}^{\infty} \phi_a(r,t)dr}{\int_{d_a - d_r}^{\infty} \phi_b(r,t)dr} = e^{\frac{(d_b - d_r)^2}{2D_b t} - \frac{(d_a - d_r)^2}{2D_a t}} &> 1 \\ \Rightarrow \frac{(d_b - d_r)^2}{D_b} &> \frac{(d_a - d_r)^2}{D_a}. \end{aligned} \tag{3.6}$$

Note that equation (3.6) is a very strong condition as it is *independent of time* which means that one node is better than the other for any expected time that it will meet the next relay. This equation defines the basic tradeoff between proximity and mobility as used in ProxiMol to identify node with better likelihood of delivery. Next we explore (3.6) for different possible scenarios that it covers.

Case I [when $d_a < d_b$ and $D_a > D_b$]

Node a that is closer to the destination has higher diffusion constant. Then from (3.6) it follows that *node a* has better likelihood of delivery for any $0 \leq d_r < d_a$ and it is reasonable to assume that next relay is between node's current position and

the destination.

Case II [when $d_a < d_b$ and $D_a < D_b$]

Node a that is closer to the destination has smaller diffusion constant. Then from (3.6) it follows that *node a* has better likelihood of delivery for any $0 \leq d_r < d_a$ if

$$\frac{d_b^2}{d_a^2} > \frac{D_b}{D_a} \because \frac{d_b - d_r}{d_a - d_r} > \frac{d_b}{d_a} \Rightarrow \frac{(d_b - d_r)^2}{(d_a - d_r)^2} > \frac{d_b^2}{d_a^2} > \frac{D_b}{D_a},$$

however, it is otherwise dependent on the distance of next relay from the destination. Note that in our analysis the next potential relay is assumed to be in the line of sight between relays nodes and the destination. In [38] Groenevelt et al. have shown expected distance of next contact for the simpler case of Brownian motion in one dimension and stationary distribution. The analysis becomes significantly more complex for nodes that do not have identical distributions and are moving in two dimensions and remains unsolved at large. Instead, we use the fact that two nodes are only compared when they are within transmission range δ of source. This implies that *node b* which is further away from destination is no more than $(d_a + 2\delta)$ distance away. Using this back in (3.6), we get

$$\begin{aligned} \frac{(d_a + 2\delta - d_r)^2}{(d_a - d_r)^2} &> \frac{D_b}{D_a} \\ \Rightarrow \frac{D_b}{D_a} &< D_{cutoff} \end{aligned} \tag{3.7}$$

where $D_{cutoff} = (1 + (2\delta/(d_a - d_r)))^2$. This implies *node a* that is closer to the destination still has better likelihood of delivery unless *node b* is D_{cutoff} times more diffusive than *node a*.

Cases I and II define the basic rule of our forwarding algorithm: *a node closer to destination has better likelihood of delivery unless it has very low diffusion.*

3.3.3 Measuring Distance and Diffusion

Thus far, we have assumed knowledge of distance of any node from the destination and its diffusion constant. Note also that we *do not require the exact location* but only the distance from the destination. Similarly, we need the ratio of diffusion constants of two nodes and *not the exact values*. This can be used to estimate which node is *more likely to reach* the next relay without actually computing the exact likelihood or probability.

3.3.3.1 Estimating distance from destination

We are interested in disconnected distance between source and destination. An estimate of this distance at some time t_0 can be used to approximate it for a later time that is not too long after t_0. Therefore, an estimate of the minimum disconnected distance from destination to source through some number of relays is used as an approximation for the distance that message will have to traverse towards destination. From (3.4), the square of expected displacement y of a node after time t is given by

$$y = \left(\int_0^\infty r\phi(r,t)dr \right)^2 = \frac{\pi D t}{2}.$$

Then for a path with k number of hops, total distance can be approximated by assumming different diffusion constants for each of the inter-contact time intervals. Since D is constant for a small time interval, we get $dy = \frac{\pi D}{2}dt$. Let D_{av} be the

average diffusion constant, f_i be the scaled diffusion of i^{th} node in the path and t_i be the time between $(i)^{th}$ and $(i+1)^{th}$ contact in the path. Then

$$\begin{aligned} y = & \int_0^{t_1} \frac{\pi D_{av} f_1}{2} dt + \int_{t_1}^{t_1+t_2} \frac{\pi D_{av} f_2}{2} dt + ... \\ & + \int_{t_1+t_2+...+t_{k-1}}^{t_1+t_2+...+t_{k-1}+t_k} \frac{\pi D_{av} f_k}{2} dt \\ \Rightarrow y = & \frac{\pi D_{av}}{2} \sum_{i=1}^{k} f_i t_i \end{aligned} \tag{3.8}$$

Equation (3.8) can be used to compute ratio of distances of two nodes from destination if value of diffusion constant or some parameter that scales linearly with it is known. It turns out that a node's rate of encounter with new nodes scales linearly with diffusion as presented in next section. Equation (3.8) also gives us a metric for the transitive utility of a node similar to what is called timer transitivity in [82, 83] and is shown effective without transitivity in [34, 39]. Roughly speaking, a node with smaller temporal distance from destination has a good likelihood of reaching it. This property has also been recently studied for mobile and online social networks in [84] as a quantitative measure of speed of information diffusion. However, our analytical result is different as time between successive contacts is scaled by diffusion of the intermediate node.

3.3.3.2 Estimating Diffusion Constant Ratios

To estimate diffusion or more precisely some parameter that scales linearly with diffusion we use the existing analysis done by Kim et al. in [49]. They show that more diffusive nodes have more contacts with new nodes per unit time. This property is same as encounter rate of a node when only first contact with a node is counted in certain time window. Equation (3.7) shows that highly diffusive nodes have better likelihood of delivery. This explains that [64, 80] are efficient in highly heterogeneous

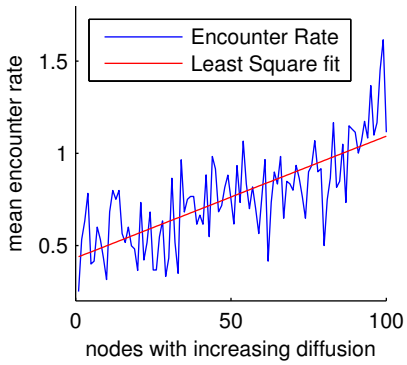

Figure 3.2 More diffusive nodes have higher encounter rate

environments because encounter rate is an estimate of diffusion. Encounter rate f is measured in a moving time window such that $f = \epsilon f_{cw} + (1 - \epsilon) f_{old}$, i.e. encounter rate is the weighted average of its current and previous value, where encounter rate in current window f_{cw} is simply the number of new contacts. Similar to [64], ϵ is fixed at 0.85 and the value of f is updated after every time-window interval.

Even though more diffusive nodes have higher rates of encounter, the linear relation is approximated after synthetic simulation of nodes with varied mobility characteristics as shown in Fig. 3.2. More specifically, truncated levy walk mobility model [73] is used with scale factor of 10 and 1 for step-size and pause-duration with α and β set to 1.7 and 0.7 respectively. To increase diffusion; pause duration, minimum step sizes and velocities of nodes are varied linearly from 10min to 1min, 5m to 25m and 1m/s to 10m/s respectively, for number of nodes 1 to 100. The simulation is run for 600min with 100 nodes using transmission range of 150m in a simulation area of 2×2 km^2. These plots indicate that that even in environments with varied mobility characteristics, encounter rate of nodes increases linearly with the diffusion of nodes on an average.

3.4 Forwarding Algorithm

Our proposed forwarding algorithm, a consequence of the analysis in Section 3.3, is described in this section. In essence, nodes are compared based on their distance from destination and their diffusion. ProxiMol is adaptive to changing network topology due to real-time distributed estimation of distance. Also, it is adaptive to changing user behavior by using a local estimate of diffusion in a moving time window. ProxiMol maintains $O(n)$ local values for rest of the network which is same as [34, 39, 82, 83] and less than $O(n^2)$ as in MaxProp [14], Prophet [59] etc.

3.4.1 Proximity

In order to estimate distance from other nodes in a distributed manner, each node keeps a timer for other nodes in networks with an update rule defined by (3.8). Let $t_a(i)$ denote the time elapsed since *node a* last encountered *node i*, where $t_a(a) = 0$. Local timer values for each node are incremented after every time unit (30s, 60s depending on devices) scaled by its diffusion as follows from (3.8). When two nodes $\{a,\ b\}$ come into contact, then $\forall i \neq b : t_a(i) < t_b(i) - t_{av}$, set $t_b(i) = t_a(i) + t_{av}$ where t_{av} is the measure of distance between two nodes when they are within each others transmission range. In [83], t_{av} is the value of estimated time that will be required for a node to physically travel the distance between two connected nodes. For our algorithm, it can be any small constant which is greater than zero but less than or equal to one time unit (increment by which local timers are updated) i.e. $t_{av} \in (0, 1]$. A small value of t_{av} helps estimate distance in disconnected space (disconnected distance) between nodes. This concept is further elaborated in evaluation in Fig. 3.8.

3.4.2 Mobility

In order to estimate diffusion, each node measures its encounter rate f_i in a moving time window of some duration. For stationary environments (behavior of nodes stays same), this duration does not have much effect as mentioned in [80], but for more dynamic environments one can use a smaller window accordingly. As mentioned in Section 3.3.1, a node's diffusion can be approximated on time scale of few steps (step length is defined in TLW [73]). This implies that our analysis on delivery likelihood is no longer applicable when a node is in close vicinity (time scale of a step or two) of destination. This threshold in time scale is denoted by t_{th}. Equation (3.7) shows that D_{cutoff} decreases with longer distance from the destination and comes close to unity. This means that nodes with slightly higher diffusion are preferred when they are at longer distances, whereas nodes that are closer to destination are preferred at shorter distances. However, actual diffusion values of nodes are not known and encounter rate is not a very accurate estimate of diffusion. Therefore, empirically, we get better results by using a constant value of D_{cutoff} (between 2 and 3) at longer distances and forwarding only to relays that are closer to destination at shorter distances (less than t_{th}).

3.4.3 Algorithm

Our final algorithm is elegantly simple and follows steps outlined below to select a relay in current neighborhood for some message. Suppose the message at *node c* is destined for *node d* and neighborhood of *node c* is N_c.

Step 1. Forward the message to $i \in N_c$ that has smallest $t_i(d)$ such that $f_i > f_c/D_{cutoff}$, otherwise

Step 2. Forward the message to $i \in N_c$ that has largest f_i such that $f_i > f_c \times D_{cutoff}$, otherwise

Step 3. Do not forward.

Note, in case of more than one node with smallest $t_i(d)$, the one with higher encounter rate is selected. Also, when $t_i(d) < t_{th}$, Step 1 is followed without any condition on f_i.

Messages are forwarded to a neighbor (with diffusion more than certain cutoff) that is closer to destination. If no such neighbor is found, then message is forwarded to the neighbor with highest diffusion (only if it's higher than D_{cutoff} times diffusion of sender).

3.4.4 Algorithm Analysis

To analyze and highlight key aspects of ProxiMol, simulations are carried out on synthetic traces as they provide a more controlled environment where node density and heterogeneity can be changed. Each node in the network generates a message every minute for a random destination in the rest of the network. Traces are generated using truncated levy walk mobility model [73] with similar setting as described in Section 3.3.3.2 except there are only two types of nodes with extreme values of the ranges shown for velocity, pause duration and step size. The number of fast moving nodes is fixed at 10 in all traces. Node density and heterogeneity are varied by changing the number of slow moving nodes from 10 to 90. Thus, a trace with 20 nodes is sparse and highly heterogenous, whereas a trace with 100 nodes is relatively dense and more homogeneous. Performance of ProxiMol is analyzed in this complete range of environments. Each point in the plots corresponds to an average of five complete runs. The confidence intervals are simply plotted as the maximum and minimum values of these runs, as they are very close to actual values (within 3%).

To describe working of ProxiMol, we first analyze its two components separately and then show the combined performance. ProxiMol is compared with TT (Timer-

Transitivity) [83], and EBR (Encounter-based routing) [64, 80]. We select TT as it shows the evolution of metrics based on time of last contact [34,39,59,82] and justifies the use of a transitive relation in comparison with other heuristics in homogeneous environments. We select EBR as rate of encounter [64], or sociability metrics of [80] are shown to achieve significant improvement in highly heterogeneous environments.

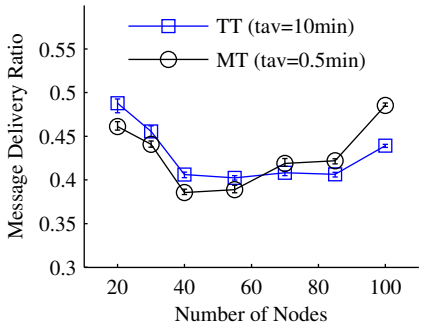

Figure 3.3 Modified timers (MT) have better delivery ratio at higher levels of connectivity

Fig. 3.3 compares message delivery ratio of timers used in [83] (denoted TT) with t_{av}=10min and slightly modified timers (MT) where t_{av} is set at 0.5min. MT has better relative performance in moderate to more dense environments (more than 60 nodes).

In Fig. 3.4 performance of encounter based routing (EBR) is compared with diffusion based routing (DBR) assuming that actual diffusion value is known. When messages are forwarded to nodes with higher diffusion (DBR), it achieves much higher delivery ratios in comparison with EBR. This is because encounter rate is not a very accurate estimate of diffusion as also evident from Fig. 3.2.

In Fig. 3.5, ProxiMol is compared with diffusion based routing (DBR) and modified timers (MT) when actual values of diffusion are assumed to be known at

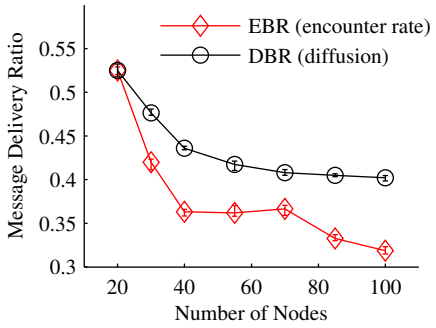

Figure 3.4 Use of actual diffusion yields up to 30% higher delivery ratio than encounter rate

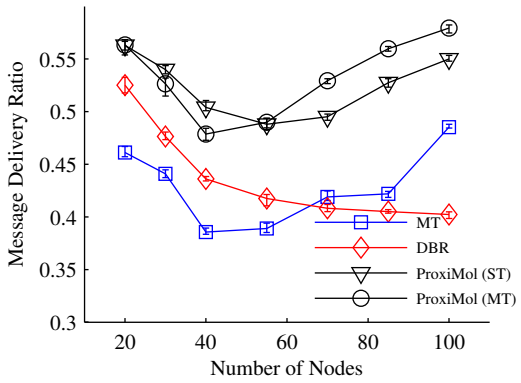

Figure 3.5 ProxiMol improves delivery ratio in environments that range from being sparse and heterogeneous (20 nodes) to dense and homogeneous (100 nodes)

each node. DBR achieves high delivery ratio in sparse heterogeneous environments whereas MT performs well in dense homogeneous environments. However, ProxiMol outperforms DBR and MT even in their respective environments and achieves 10-20% higher delivery ratio in the complete range of environments.

However, in real traces, nodes do not know their diffusion values. From simulations it is found that scaled timers used by ProxiMol are not efficient when actual encounter rate is used instead of diffusion. Therefore, we compare ProxiMol when it uses scaled timers (ST) with known diffusion values and modified timers (MT). The difference is that in ST local timers are updated based on diffusion/encounter rate

of node, whereas in MT timers are simply updated by a unit time interval. As seen from Fig. 3.5, ProxiMol still adapts quite well in all environments when MT is used to estimate distance.

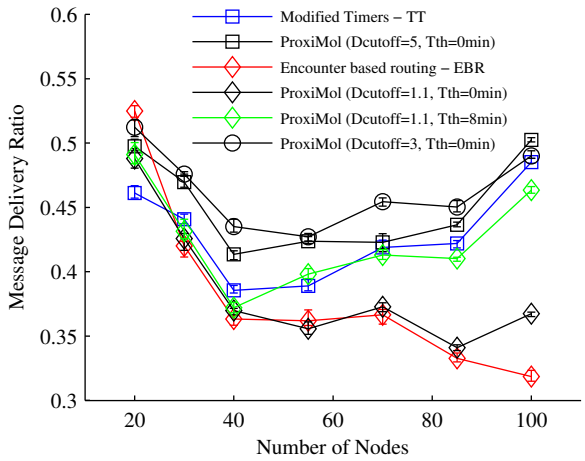

Figure 3.6 ProxiMol's performance is close to Encounter Based Routing (EBR) at low D_{cutoff} values and close to Modified timers (MT) at high D_{cutoff} values

Fig. 3.6 explores the effect of D_{cutoff} and t_{th} when ProxiMol uses MT to estimate distance and actual encounter rate to compare diffusion of two nodes. ProxiMol behaves similar to EBR at small values (D_{cutoff}=1.1) whereas it is closer to MT at higher values of diffusion cutoff (D_{cutoff}=5). Generally values between 2 to 3 yield good results in the complete range of environments. The effect of t_{th} is more visible in environments with moderate to higher levels of connectivity. As seen from Fig. 3.6, at D_{cutoff}=1.1, ProxiMol has upto 30% higher delivery ratio at t_{th}=8min as compared to t_{th}=0min. Basically, t_{th} allows the message to reach destination if a connected path is available in close vicinity of the network. When it is set at zero, a message residing on a mobile node may not be forwarded (due to conditions on encounter

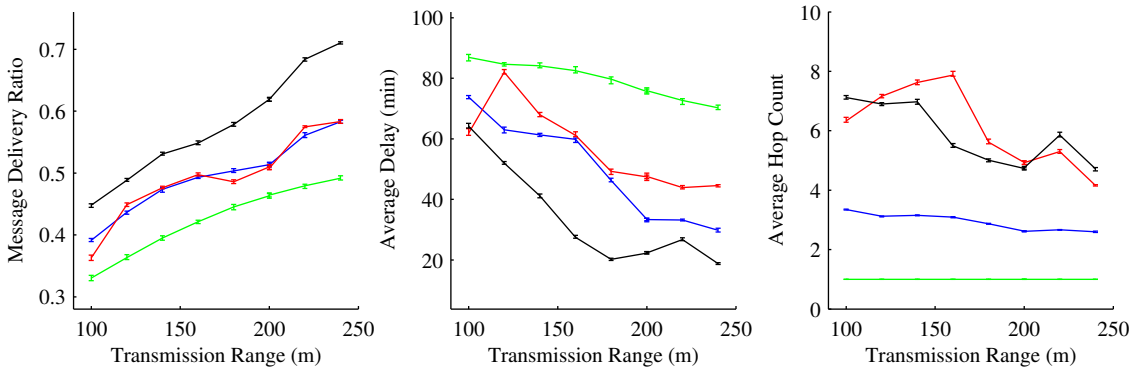

Figure 3.7 KAIST: (a) Delivery Ratio, (b) Delay, (c) Hop Count of forwarding schemes [ProxiMol (black), TT (blue), EBR (red) and Direct (green)]

rate) even when a connected path to destination exists. Therefore, higher values of t_{th} yield better delivery ratio at moderate to higher levels of connectivity.

3.5 Simulation and Analysis

3.5.1 Data Set

Additional simulations studies are carried out on *real mobility traces* from three different environments (Disney World Orlando, KAIST university campus, and State Fair) rather than any synthetic mobility model. Real traces allow standardized comparison of forwarding metrics, and do not have any artificial features that can alter actual performance results. Details on collection of these traces are available in [73]. We overlay (to increase contact opportunities) Disney traces onto a 2×2 km^2 wrap-around simulation area and truncate all trace logs to the time until which a suitable number of users record their location. Total number of track logs used and their truncation times are provided in Table 3.3.

Table 3.3 Data set used for simulation

Traces	**#tracklogs**	**Trunc. Time**	**Setting**
KAIST	92	250 min	Original
Disney World	39	250 min	Overlay
State Fair	19	89 min	Original

3.5.2 Setup

Each node in the network generates a message every minute for a random destination in the rest of the network. Nodes update their neighborhood every 30s in a distributed manner (a new one hop path might be discovered after two updates instead of one, depending on the order of updates). Only single copy of a message is used in forwarding to see the impact of efficient estimation of delivery likelihood alone, independent of any specific kind of controlled replication that can possibly be used [14, 64, 82]. The design goal in this dissertation is to identify nodes with best likelihood of delivery. Once these are identified, ProxiMol can be combined with other frameworks that do constraint optimization [8]. For this reason, infinite bandwidths and buffer space availability are assumed. Algorithm parameters are: encounter rate window is 10min for EBR and ProxiMol, t_{av}=0.5min, D_{cutoff}=2.5, t_{th}=8min for ProxiMol and t_{av}=10min for TT.

3.5.3 Analysis

To demonstrate effectiveness of ProxiMol in estimating nodes with better likelihood of delivery, we show that it achieves higher delivery rates, and significantly reduces the average delays in a range of realistic environments with different settings for level of connectivity. To account for heterogeneity, real mobility traces from different settings are used. Since real mobility traces are used, different levels of con-

nectivity are investigated by varying transmission range of the nodes from 100m to 240m in steps of 20m. Fig. 3.7(a) shows that ProxiMol increases the delivery ratio by 12% at lower level of connectivity to 24% at higher level of connectivity in comparison with TT and EBR. Results using direct delivery (source waits until a direct contact with destination) are also included for reference as it is similar to Spray and Wait [81] without use of replication.

The better relative performance of ProxiMol at higher levels of connectivity is due to estimates of disconnected distance. This is explained with the help of Fig. 3.8 that shows the timer values for some destination in TT and ProxiMol at intermediate nodes. Timer values are different due to value of t_{av} [Section 3.4.1]. Based on the timer values, *white node* will forward the messages to *blue node* (smaller actual distance) in TT but use the connected shortcut (*black nodes*) in ProxiMol. This is precisely how ProxiMol exploits the partial paths in moderately connected networks. Note that a small, fixed value of t_{av} is also useful in finding shortest paths in a connected subset of nodes.

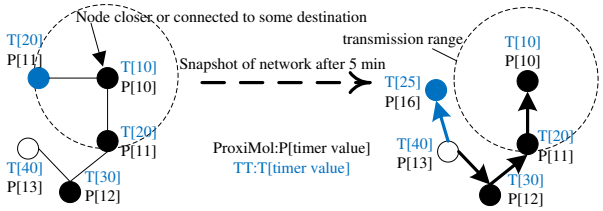

Figure 3.8 Relay selection at *white node* in ProxiMol and TT. TT prefers a node with smaller physical distance from destination whereas ProxiMol prefers a node that is further away but has a connected path to destination

Fig. 3.7(b) and 3.7(c) show the average delay and hop count in KAIST. ProxiMol significantly reduces the average delay (by over 50% at moderate levels of connectivity) but takes more number of hops than TT. This is because, in ProxiMol, messages

do not wait on a single relay (like they do in TT) when they find a more mobile node in the vicinity. Exploiting more mobile nodes results in earlier delivery of a message but incurs additional hops. This same effect is visible in Fig. 3.9(c) and 3.10(c).

Performance results in Disney World Orlando and State Fair are summarized in Fig. 3.9 and 3.10. ProxiMol improves message delivery ratio by around 10% in both cases (Fig. 3.9(a) and 3.10(a)). ProxiMol reduces delay by up to 50% at higher levels of connectivity in State Fair (Fig. 3.10(b)) but only around 5% in Disney world (Fig. 3.9(b)). Note that the delay and hop count are measured *only* for successfully delivered messages. An efficient forwarding scheme can still have higher average delay and hop count when it delivers extra messages at higher delays using more number of hops that other schemes simply fail to deliver. This effect is clearly visible in Fig. 3.9(b) where the least efficient scheme in terms of delivery (direct delivery) incurs the minimum delay. Therefore, in comparing delay and hop count, improvement in delivery ratio must be considered. This means that ProxiMol is still more efficient when it shows similar delays but much higher delivery ratio at transmission range of 150m in Fig. 3.9(a) and (b).

ProxiMol improves performance by exploiting the tradeoff between mobility and proximity. ProxiMol exhibits following characteristics in different environments:

- Pure homogeneous environments – behavior is similar to that of TT with the added advantage of using disconnected distance.
- Specific heterogeneous environments – similar to EBR with the added advantage of exploiting multi-hop paths that transport the message closer to destination.
- Mixed environments – uses nodes with better mobility when available, uses partial paths that bring the message closer to destination and exploits proximity of location among nodes with similar mobility characteristics.

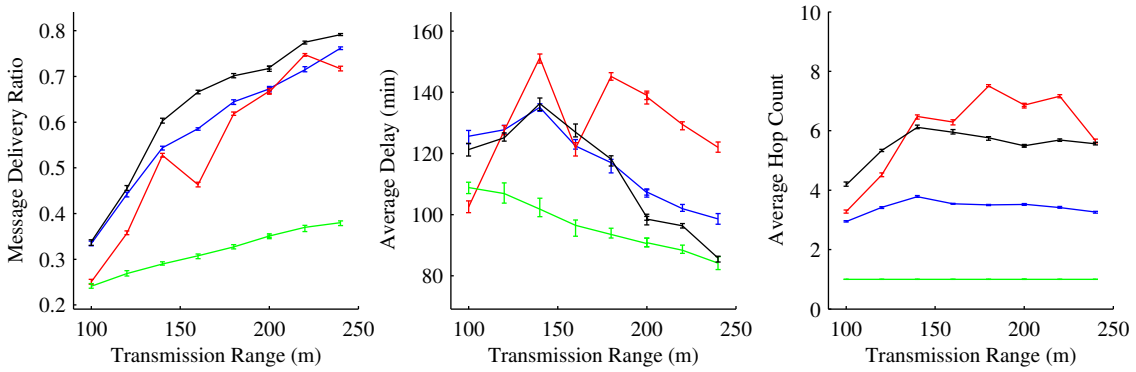

Figure 3.9 Disney World: (a) Delivery Ratio, (b) Delay, (c) Hop Count of forwarding schemes [ProxiMol (black), TT (blue), EBR (red) and Direct (green)]

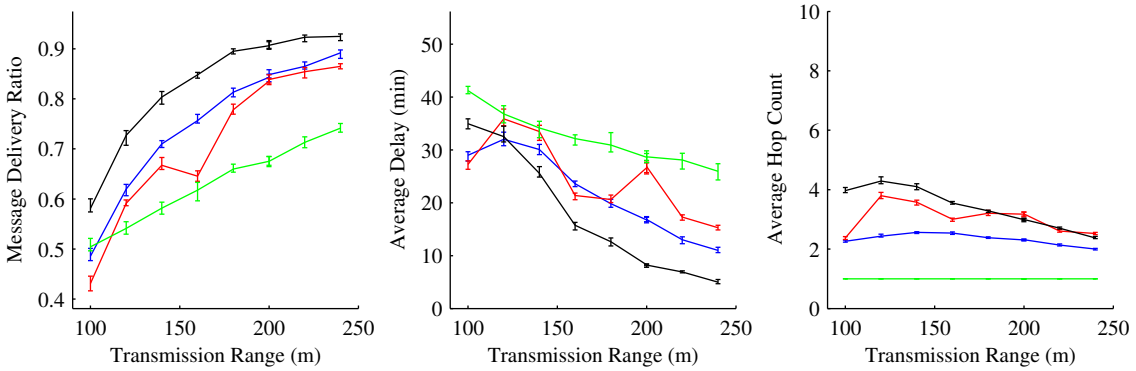

Figure 3.10 State Fair: (a) Delivery Ratio, (b) Delay, (c) Hop Count of forwarding schemes [ProxiMol (black), TT (blue), EBR (red) and Direct (green)]

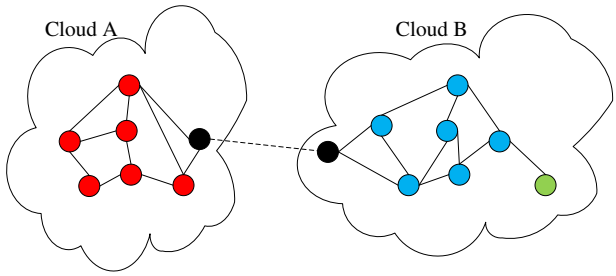

Figure 3.11 Temporary disconnection between two node clouds

In [42], Heimlicher et al. have shown that in partially-connected networks, intermediate routing (routing to the point of last disconnection) is superior to source routing (forwarding on a complete path). Fig. 3.11 shows two node clouds with black

nodes as the point of last disconnection. In [20], Cai et al. show that in some scenarios, the node with smallest timer (black nodes) may have the least likelihood of connecting back again. TT forwards all messages destined for other cloud to the black node which may have the least likelihood of delivery. In contrast, ProxiMol also looks in the neighborhood of black node for more mobile nodes that can take messages to the other cloud. Intuitively, proximity is established by the timer value but neighborhood is explored for more mobile nodes.

3.6 Summary

In this chapter, ProxiMol, an efficient and adaptive forwarding scheme, is presented to improve message delivery ratio and reduce average delay in an opportunistic network. Based on our analysis and simulation studies, we conclude that *mobility* (measured by diffusion of a node) plays a central role in a highly heterogeneous network environment while a node's *proximity to destination* is useful in relatively homogenous settings in improving delivery ratio. Using statistical properties of real mobility traces, a model is proposed to approximate a node's location over time using a locally measured value of its diffusion. This model is used to define the key tradeoff between proximity and mobility to compare delivery likelihood of nodes. ProxiMol shows marked improvement in message delivery ratio, while at the same time reduces average delay in a range of environments with varied mobility characteristics and different levels of connectivity. To the best of our knowledge, this is the first work that derives a forwarding metric based on power-law distributions of inter-contact times, flight lengths and pause durations. These characteristics have been shown to be statistically significant in real mobility traces [73].

In addition, it is shown that partial paths towards destination in networks with moderate level of connectivity can be exploited to achieve significant improvement

using the concept of disconnected distance. Simulations on real mobility traces from three different environments are provided to validate our results.

CHAPTER 4

SERVICE COMPOSITION IN OPPORTUNISTIC NETWORKS

In the previous chapter, we provided a forwarding scheme that can be used to transfer data betweeen devices. Using a forwarding scheme, in this chapter, we provide a distributed algorithm for composition of services that are available on different devices in an opportunistic network.

4.1 Introduction

When two devices are within communication range, there exists an opportunity to exploit each other's services. If the contact period is sufficiently long, nodes can utilize each other's resources to execute remote services. On the other hand, if the contact time is insufficient, service parameters can be exchanged during the contact, while service itself can be executed offline and service outputs transmitted to the requester using opportunistic contacts. In Passarella et al. [68,69], multiple copies of a request are spawned to minimize delay in service execution and to increase robustness but only one-hop communication (i.e. a requester waits until a direct contact with service provider) is used. Also, a few other works propose efficient schemes and use multi-hop paths for service discovery and invocation in opportunistic networks by using a set of proxies [52] or location of the user [79]. However, composition of multiple services is not considered in these works. For composition of services, a number of middleware frameworks and architectures are proposed but these do not discuss the actual forwarding mechanism [22, 32, 48]. The use of multi-hop paths to send service requests and receive service outputs between nodes is considered in [33,86]

by modeling the expected colocation time between nodes based on past interactions. However, service composition fails if participants do not remain directly connected. In contrast in this dissertation, service composition is performed using multi-hop paths that can relay information to the service providers and then relay it back to the service requestor even when an end-to-end connected path does not exist. Furthermore, alternative paths are considered to increase the success of composition.

The problem of service composition is different from traditional routing in two key aspects: i) service load at destination needs to be taken into account; and ii) future forwarding path needs to be considered for the remaining services that need to be composed in a sequence. Our algorithm selects a service sequence set by taking into account the service load and temporal distances between nodes (temporal distance provides a measure of relative location of other nodes). These values are measured in a distributed manner by using only opportunistic contacts. The elegance of our solution lies in the fact that while a particular service sequence is selected by considering the temporal distance between devices, the actual routing scheme used to forward service request and parameters can be different. For example, to meet a service request of *encrypting and compressing* a file, our algorithm chooses a sequence of services (e.g., an encryption service and then a compression service) that can be composed such that these services are provided at devices in proximity of the service requester. [1] Our proposed service composition uses an underlying routing scheme to forward service request to the device providing encryption, encrypted result to the device providing compression and lastly to transmit final result back to the service requester.

[1]Specifically, devices whose distance from the requester (a proxy for the time to reach them thorough a multi-hop path) and load is such that the service can be provided in a short amount of time. Our simulation results shows that our heuristic is effective in finding the best solution.

4.1.1 Contributions

Extensive simulation results are presented to demonstrate the performance of the proposed method in terms of service composition success, composition length, number of hops, and delays. The performance is analyzed for a range of service densities, number of nodes, service request rates, request timeout durations, and routing mechanisms. It is shown that: i) composition is better than searching for the exact single service match; ii) use of multi-hop paths improves performance (service completion rate and delay) significantly as compared to one-hop direct forwarding; and iii) using only local knowledge (collected from opportunistic contacts) about load at and temporal distance from other nodes, performance close to a centralized system (exact load and temporal distances between nodes are known) can be achieved. Additional key contributions made in this dissertation are as follows:

1. Scope and applicability of our service composition algorithm by considering a range of mobility characteristics - movement in communities, clustered environments, and uniformly distributed mobile users [Section 4.6.1].
2. Performance guarantees - relationship between estimated cost in the service graph and actual delay to complete the service composition request [Section 4.6.3].
3. Sensitivity to composition length and service distribution [Section 4.6.4].

In the rest of the chapter, the terms *devices* and *nodes* are used interchangeably. This chapter is organized as follows: Section 4.2 provides the system description. Service composition mechanism and algorithm is described in Section 4.3 followed by modeling of temporal distance and service loads in Section 4.4. Performance of service composition algorithm is evaluated in Section 4.5. Specific characteristics of the algorithm and sensitivity to key factors are analyzed in Section xcharcal. Section 4.7 provides summary of the chapter with directions for future work.

4.2 System Description

4.2.1 Service Model

We employ a service graph to represent services, their inputs and outputs, and concatenation of services based on their input requirements. This can be done by using service ontologies and semantics in a hierarchical manner [48]. For simplicity, we represent a service based on its input and output types to generate a service graph. Two services can be composed if output type of one is same as required input to the other. For n_d input/output types, possible services are of type s_{xy} such that input type $x \in [1, n_d - 1]$ and output type $y \in [x + 1, n_d]$. This representation illustrates services with different levels of functionality. A service s_{xy} is of functionality k if $y = x+k$. Suppose, s_{12} and s_{23} represent services *decompress* and *decrypt* respectively, then service s_{13} represents a service of higher functionality that can perform both decryption and decompression. A service with a higher functionality such as s_{14} provides same transformation that is otherwise possible by composition of two or more services with lower functionalities (e.g., $\{s_{12}, s_{23}, s_{34}\}$, $\{s_{12}, s_{24}\}$ or $\{s_{13}, s_{34}\}$). Services with varying functionalities are used to demonstrate the tradeoff between selecting an exact service (that can be further away in the network) versus composition of smaller services (that might be available in proximity of service requester). To account for task and device heterogeneity, we assume that execution time of each service is distributed exponentially on the devices. The performance of service composition is explored by varying the repetition of each service i.e. the number of nodes providing that service.

4.2.2 Mobility Model

Real mobility traces are known to have a power law distribution of Inter-Contact Times (ICTs) [23] in contrast to previously assumed exponential distribution. Since random waypoint and random direction models lead to exponential Inter-Contact

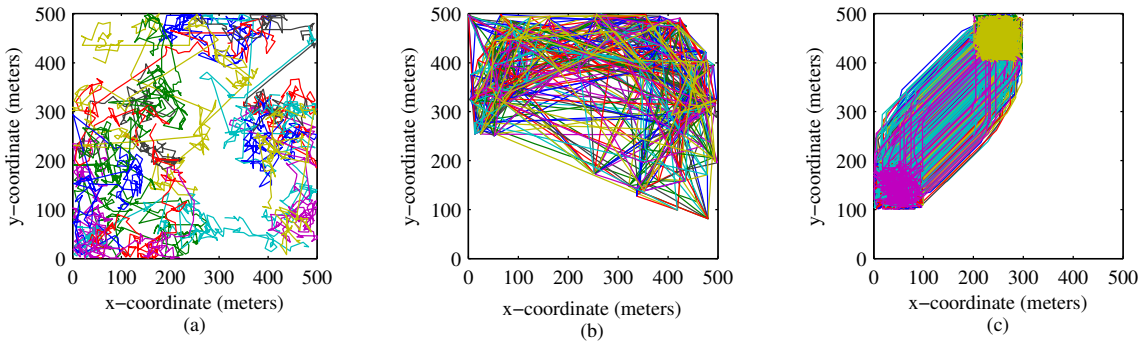

Figure 4.1 Trajectories of 20 users in (a) Levy walk, (b) SLAW, and (c) HCMM mobility models.

Times [77], they are unsuitable to represent nodes' mobilities. In addition, it has been shown that real mobility traces have power-law flight-lengths and pause-time distributions [73]. Therefore, in our analysis and simulation, we generate synthetic traces using Levy walk mobility model [73] and SLAW mobility model [53]. Levy walk and SLAW represents a more realistic scenario in comparison with [69] where each node is equally likely to meet any other node. In addition, we also evaluate performance using the HCMM mobility model [11] that models spatial and temporal properties of human mobility by also taking into account that user mobility is driven by existing social relationships between them.

Mobility plays a central role in the selection and forwarding of service requests in an opportunistic network. These models provide a wide range of mobility characteristics to thoroughly evaluate performance of service composition: HCMM - movement in communities; SLAW - a clustered environment; and Levy walk - uniformly distributed mobile users. These characteristics are also reflected in Figure 4.1 that shows trajectories of 20 users under these models. Flight lengths have power law distribution and user mobility is spread in the entire area under Levy walk mobility whereas only specific sites (called waypoints [53]) are visited under SLAW mobility.

As a result user movement is more evenly spread in Levy walk [Fig. 4.1a] and contains clusters in SLAW [Fig. 4.1b]. Fig.4.1c shows the extreme case where users only move between two communities where this movement is controlled by the parameter of rewiring probability in HCMM [11].

Network environment is made heterogeneous by varying the mobility characteristics of nodes. Based on levy walk, though each node always follows the same step length, pause time distribution and speed for itself, these parameters are varied across the network for other nodes. This is reflected in Table 4.2 that shows the values for slow and fast moving nodes. In addition to synthetic traces, results on real mobility trace from a State Fair [74] are also analyzed. The corresponding results support applicability of our proposed algorithm in homogenous (nodes with similar mobility characteristics) as well as heterogeneous (nodes with varied mobility characteristics) environments. This dissertation focuses on modeling and analysis of service composition in physical proximity of a user.

4.2.3 Forwarding Schemes

There are several forwarding schemes proposed in the literature on opportunistic networks. These are discussed in Section 2.2. The following forwarding schemes have been found to have good performance in dynamic environments: i) use of timer transitivity (TT) [83] or temporal distance between nodes in homogeneous (nodes with similar mobility characteristics) environments; and ii) use of encounter rate (EBR) [64, 80] in heterogeneous environments (nodes with varied mobility characteristics). In addition, we have proposed a modified version of timers (MT) in Chapter 3 that works efficiently over small forwarding paths. In our simulation studies, we compare performance of the proposed selection and forwarding mechanism for service composition under 4 forwarding schemes - direct or one-hop, TT, EBR, and MT.

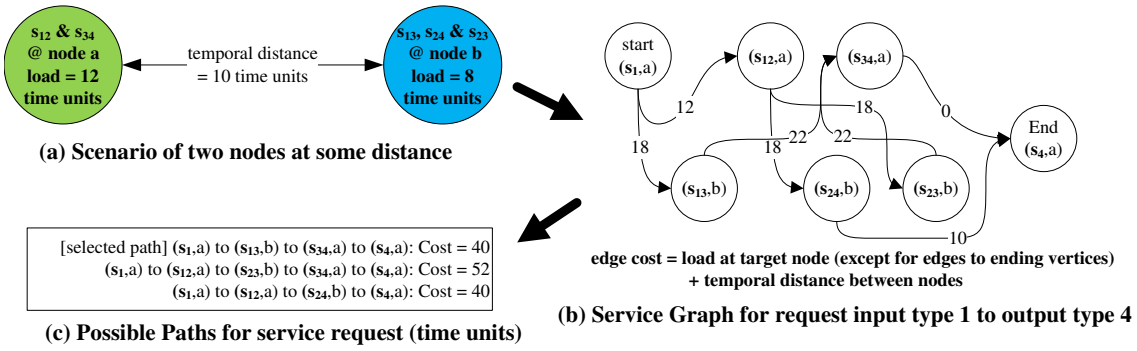

Figure 4.2 Overview of Composition Selection Algorithm: (a) an example scenario, (b) construction of service graph, and (c) possible paths for request completion

Direct forwarding belongs to the class of one-hop routing schemes whereas, TT, EBR and MT belong to the class of multihop routing schemes. TT and MT are representative for homogeneous network environments whereas EBR is representative for heterogeneous network environments.

4.3 Service Composition Algorithm

In this section we provide an overview of the service composition algorithm. Each device maintains a service graph and the link costs on the service graph are updated based on temporal distance between nodes and their current service loads. A particular service composition is selected based on the shortest path in the service graph. Overview of the key algorithm steps is also provided in Figure 4.2. Resource modeling for temporal distance and load as wells as methods for estimation are provided later in Section 4.4.

4.3.1 Service Graph

Each device maintains a local service graph $G = (V, E)$ based on its view of rest of the network. A simple case is described in Figure 4.2, where services provided

by two devices a and b are used to construct a service graph. The number of devices, services (including repetitions) and input/output types are represented by N, N_s and n_d respectively. The service graph G has two types of vertices $V = \{V_1, V_2\}$, where a vertex $v \in V_1$ is a device and service pair such that $|V_1| = N_s$, and vertex $v \in V_2$ is a device and input/output type pair such that $|V_2| = n_d$. The two types of the vertices are shown in Figure 4.2b: i) (s_{12}, a) is a vertex of type V_1, represents that service s_{12} is provided at node a; and ii) (s_1, a) is a vertex of type V_2, represents that there is an input/output of type 1 (denoted s_1) entering/exiting an application at node a. Vertices of type V_2 are only maintained for the device constructing the service graph.

A directed edge (u, v) between two vertices u and v exists if (i) input/output type of first vertex $u \in V_2$ is same as service input of second vertex $v \in V_1$ e.g., $(s_1, a) \rightarrow (s_{12}, a)$; (ii) service output of first vertex $u \in V_1$ is same as service input of second vertex $v \in V_1$ e.g., $(s_{12}, a) \rightarrow (s_{24}, b)$; or (iii) service output of first vertex $u \in V_1$ is same as input/output type of second vertex $v \in V_2$ e.g., $(s_{24}, b) \rightarrow (s_4, a)$. The cost of an edge in cases (i) and (ii) is the sum of temporal distance between devices of corresponding vertices and the load at the device of second vertex. For example, cost of $(s_{12}, a) \rightarrow (s_{24}, b)$ is 18 which is sum of temporal distance from a to b (10) and load on b (8). However, the cost of an edge in case (iii) is simply the temporal distance between devices of corresponding vertices as it is the cost of routing final results back to the service requester. Vertices of type V_2 are needed to take into account the temporal distances: (a) from the service requestor to the node providing the first service; and (b) from the node providing the last service to the requestor.

4.3.2 Algorithm

Our final algorithm is to compute a shortest path on a service graph based on a service request (input type and desired final output type). From the collected

information about service loads, temporal distances, and services provided at the neighboring nodes, a device creates the service graph. In Figure 4.2, a simple case is described for a service request from input type 1 to output type 4 at device a. Using the service load and temporal distance estimate, the edge cost is computed from each service output to compatible service inputs. Using Dijkstra's algorithm, shortest path from the starting vertex (s_1, a) to ending vertex (s_4, a) is computed as shown in Figure 4.2. In this case the shortest path is $(s_1, a) \rightarrow (s_{13}, b) \rightarrow (s_{34}, a) \rightarrow (s_4, a)$ i.e. the next service s_{13} is to be run at device B. This information is given to the forwarding algorithm to route service request from device A to device B. After execution of this service in the sequence, the device hosting the service (device B) finds the next service in the path by re-computing the shortest path for pending request (input type 3 to output type 4) and gives the destination address to the underlying forwarding scheme. Service execution and re-computing the shortest path for pending request is done until the final results are routed back to the service requestor. The path is re-computed after execution of each service in the sequence as the network topology can change in that duration to provide alternate paths that are more feasible. However, one may choose to follow the same path computed at the service requestor if the network topology does not change very fast.

The network overhead in terms of data shared upon each contact between two nodes is only $O(N)$ values. These values comprise of one estimate for load and temporal distance for each device in the network. The service graph has a total of N_s+n_d vertices. The computational complexity of updating the service graph requires updating $O(N^2)$ values (of edge cost) rather than $O((N_s+n_d)^2)$ values because the cost of edges from services on one device to services on another is same. Also, Dijkstra's algorithm has a computational complexity of $O((N_s + n_d)^2)$. One run of Dijkstra's Algorithm finds shortest path from one input type to all possible output types i.e.

it finds composition paths of all service requests that have the same input type. Thus, at most, a node only requires to run Dijkstra's Algorithm n_d times to compute composition path for any number of pending service requests. To control network overhead and computational complexity in a large network, the size of graph can be reduced by retaining information of only those devices that are in close vicinity.

4.4 Resource Modeling and Parameter Estimation

4.4.1 Disconnected Paths

In opportunistic networks, nodes providing a service may not be directly connected to the service requested. This leads our investigation into modeling the physical distance between nodes when these are devices carried by people. In [50], Kim et al. show that, under Levy walk, the order of magnitude of typical displacement of a mobile node is given by $r = t^{\gamma/2}$ where r^2 (denoted $M(t)$ in [50]) is the mean square displacement of a node at time t from its position at time 0 and $\gamma \in [0.5, 2]$. Then, the distance d between two nodes when they start from same position at time 0 is given by $d = 2r\sin(\theta/2)$ where $\theta \in [0, 2\pi]$ is the angle between their displacements and is uniformly distributed in the range. Therefore, expected displacement is given by

$$E[d] = 2r\frac{1}{2\pi}\int_0^{2\pi} \sin(\theta/2)d\theta = \frac{4r}{\pi} = 4t^{\gamma/2}/\pi. \tag{4.1}$$

Interestingly the physical distance between nodes is related to the separation in time i.e. time since they moved out of each other's transmission range. Also, nodes can share information about separation in time to create a local view of the entire network (i.e. distance / separation in time from every other node in the network). A mechanism to find and share such values (separation in time) has been proposed in [83] and the final computed values for separation in time is called (and is equivalent to)

shortest temporal distance. Several works (including our work on data forwarding [Chapter 3]) have used the intuition behind this idea to approximate the physical distance between nodes and characterize the temporal distance [20,34,39,83,84]. Besides its relation to the physical distance, an alternate definition of shortest temporal distance between two disconnected nodes is that it is the minimum time in which some sequence of contacts can relay information from one node to the other [84]. This is described in the next section.

4.4.2 Shortest Temporal Distance

In essence, shortest temporal distance (t_{ij}) provides the minimum time (using epidemic forwarding) it would take for some information to travel from node i to node j. Therefore, to compare efficiency of an underlying forwarding scheme to send requests to different nodes in network, we simply compare this lower bound on the propagation time. For example, node i can select a service provider node j or k based on which of these has smaller distance (t_{ij} or t_{ik}). Since, no centralized infrastructure exists in an opportunistic network to provide t_{ij} and t_{ik}, the approximations t_{ji} and t_{ki} are used as they can be measured at node i in a distributed manner by using simple timers [Section 4.4.4].

We analyze the shortest temporal distances maintained at nodes in a real mobility trace from State Fair. In this chapter, the term temporal distance always refers to the shortest temporal distance. It is found that even though actual nodes in the shortest path may be different, the temporal distance t_{ij} is approximately same as t_{ji}. This is also validated by the fact that temporal distance relates to the physical distance between nodes. As the distance between nodes i and j is the same in both directions, the temporal distances are also assumed to be equal in both directions. This is demonstrated in Figure 4.3. We measure the absolute difference $|t_{ij} - t_{ji}|$

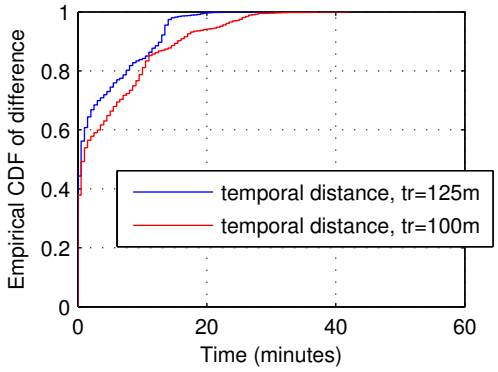

Figure 4.3 CDF of difference in temporal distance $|t_{ij} - t_{ji}|$ for all node pairs i, j at different transmission ranges (tr)

for all node pairs i,j after every 30 sec in a 90 minute duration of State Fair mobility trace [74] with 18 users. Figure 4.3 shows the CDF of these values. Since real mobility trace is used, different levels of connectivity are investigated by changing the transmission range tr from 100m to 125m. At higher level of connectivity (tr=125m), differences between temporal distances are smaller than at lower level (tr=100m). Also as shown in Figure 4.3, differences are less than 5 minutes for 50% and less than 10 minutes for around 80% of pairs at all times which justifies the use of approximation t_{ij} for t_{ji}.

4.4.3 Service Loads

Each node provides one or more services, and maintains a FIFO queue for all service requests. Thus, even though a requester may be connected to a service provider, the request may take a long time to complete because service provider is busy. Therefore, one needs to take into account the current service load at the destination in order to compose services in minimal time. This can be done by sharing service load of all nodes in a distributed manner [Section 4.4.4]. Our service selection algorithm takes into account the service load at destination together with the

temporal distance estimate. Though, statistical mechanisms can be used to predict the future service loads and provide better estimates, our current analysis uses a moving time-window average for the load estimate l at each node with a decaying factor χ of 0.5. Load in current window l_{cw} is calculated by the total number of pending requests multiplied by the expected service execution time. Then estimate of load is updated by the following rule: $l = \chi l_{cw} + (1 - \chi) l_{old}$ where l_{old} is the load value l in previous time window. We use a time window of 30s in our analysis.

4.4.4 Estimation of Temporal Distance and Load

We are interested in temporal distance between nodes. An estimate of this distance at some time t_0 can be used to approximate it for a later time that is not too long after t_0. In order to estimate temporal distance from other nodes in a distributed manner, each node keeps a timer for every other node in the network using an update rule defined below. Nodes also share their load values with rest of the network. This is in contrast with centralized estimates of load that are used in [69]. Each node maintains a load value for other nodes in network. Let $t_a(i)$ denote the time elapsed since node a last made contact with node i, where $t_a(a)$ is always set at zero. Also, let $l_a(i)$ denote the node a's estimation of load at node i. Local timer values for each node are incremented after every time unit (e.g., 30s, 60s etc. depending on devices). When node a comes into contact with some other node b, it updates its timers and loads estimates for all nodes from whom node b has a smaller temporal distance. This rule is defined as follows: $\forall i \neq a$ if $t_b(i) < t_a(i) - t_{av}$, then,

$$\begin{aligned} t_a(i) &:= t_b(i) + t_{av} \\ l_a(i) &:= l_b(i) \end{aligned}$$

where t_{av} is the measure of distance between two nodes when they are within each others' transmission range. Every node performs this update when it comes into contact with another node. Note that $t_a(i)$ is same as temporal distance t_{ia} and is used to approximate t_{ai} at node a as described in Section 4.4.2. The value of t_{av} is a small constant greater than zero but less than or equal to one time unit (increment by which local timers are updated) i.e. $t_{av} \in (0, 1]$. Note that it's a different definition of t_{av} from that in [83] where the value of average time required to travel between the two nodes is included. A small value of t_{av} is used in our experiments as we are interested in the time a request will take in forwarding and not the physical distance between nodes. Value of $t_{av} > 0$ creates a gradient in a connected subset of nodes such that forwarding paths with smaller number of hops are selected. Based on this smaller value of t_{av} we have proposed a forwarding scheme modified timer (MT) [Chapter 3] in which messages are forwarded to nodes with smaller timer from the destination. MT is used as the default forwarding scheme in all simulations and is found to perform efficiently when temporal distances are small ($<$ 15min).

4.4.4.1 Service Advertisement and Controlling Distribution Radius

Each node keeps track of its temporal distance from all other nodes for the service composition algorithm. To reduce network overhead, temporal distance values of only those nodes are stored that are in close vicinity (e.g. nodes with temporal distance less than 15 minutes). The exact threshold value for retaining temporal distance values of other nodes depends on the application requirement in terms of how much delay can be tolerated. This makes our approach scalable for large networks, as nodes have a direct mechanism to filter unnecessary information about devices that are further away. Thus, each time two nodes come into contact, they only update the timers and service load estimates for those remaining nodes in network whose

Table 4.1 Settings for different levels of awareness

Level	$\widehat{t_a(i)}, \widehat{t_i(a)}$	$\widehat{t_i(j)} : i, j \neq a$	$l_a(i)$
minimal	1	1	0
local	$t_a(i)$	$t_a(i) + t_a(j)$	$l_a(i)$(delayed)
global	$t_a(i)$	$t_i(j)$(delayed)	$l_a(i)$(delayed)
perfect	$t_a(i)$	$t_i(j)$	$l_a(i)$

temporal distances (implied proximity of location) are smaller than the threshold value (which is suitable for application requirements). Similar timers can directly be used to control epidemic forwarding of description of services that other nodes provide to a limited space. As network size grows, nodes only maintain a fixed set of values to efficiently select and forward service requests.

4.4.5 Levels of Awareness

Note that even though any forwarding scheme can be used to forward service requests efficiently, the selection of service set (and consequently the destination nodes for forwarding) still needs to take into account the likelihood of delivery. Even though our algorithm is independent of any forwarding scheme, selection of a particular service set has a significant impact on performance (service completion rate and delay) of any forwarding algorithm. The selection of a particular service set can be made with different levels of awareness about rest of the network. More specifically, the edge cost between vertices of service graph is based upon the knowledge of temporal distance and service loads. These levels are summarized in Table 4.1 for node a where $l_a(i)$ represents estimate for load of *node* i at node a and $\widehat{t_i(j)}$ represents estimate for temporal distance $t_i(j)$ at node a. Note that, timers maintained by node a only provide $t_a(i)$, therefore, different levels of awareness are used to estimate $t_i(j)$ for $i, j \neq a$.

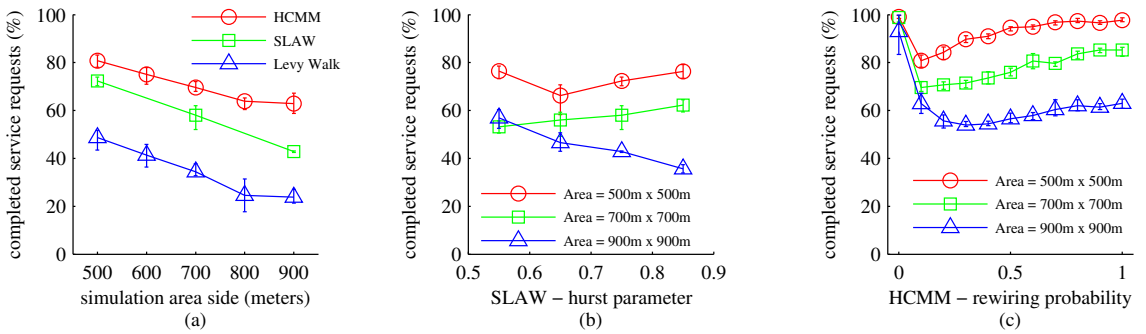

Figure 4.4 Performance of service composition under varying mobility parameters in (a) Levy Walk, (b) SLAW, and (c) HCMM mobililty models.

At the very basic level it is assumed that a node knows about services being provided in the environment but does not have an estimate of its temporal distances from particular devices. This is reflected as *minimal* in Table 4.1 as it requires no housekeeping. In this case, a node randomly selects the device providing the required service(s) for composition.

The next level of *local* knowledge is by use of timers as described in Section 4.4.4. A node has knowledge of its temporal distance from other nodes in the network (e.g., $t_a(i)$ for node a). A node also knows about the latest service load at other nodes, which is forwarded to it in a distributed manner [Section 4.4.4] and therefore incurs some delay. This level requires exchanging $O(N)$ values $(t_a(i), l_a(i)\ \forall i$ at node $a)$ per contact. In order to update link cost between services provided at other nodes, a node needs to be aware of temporal distances between other nodes (i.e. $t_i(j) : i, j \neq a$). However, this information is not available in *local knowledge*. Therefore, a node uses its own timer values for approximation as shown in Table 4.1. For example, node a uses $(t_a(i) + t_a(j))$ as an approximation for temporal distance $t_i(j)$ because $|t_a(i) - t_a(j)| < t_i(j) < t_a(i) + t_a(j)$. Even though this approximation (an upper bound on $t_i(j)$) is not accurate, it still gives an idea of proximity of a device. Later

we show that the performance results using local knowledge are fairly close to ones achieved by perfect knowledge.

The next higher level of *distributed global* knowledge is when a node also receives timers of other nodes in a distributed manner i.e. the estimates of temporal distances between all nodes which requires exchanging $O(N^2)$ values ($t_i(j), l_a(i)\ \forall i, j$ at node a) per contact. However, this information about temporal distance between other nodes is still not up-to-date because of propagation delay. Therefore, we further make comparison with a *perfect* system which assumes centralized knowledge of temporal distances and service loads i.e. all information is available to nodes without any delay.

4.5 Performance Evaluation

Performance of service selection algorithm is evaluated for different types for levels of awareness, forwarding schemes, repetition of services, request rates, and node densities. Extensive simulations are run on real as well as synthetic mobility traces. The real mobility traces have been collected from participants that carry GPS receivers which log position at 30 second intervals in a State Fair [74]. In order to make a comparison with suitable number of nodes, track logs by same user on different days are considered to be a separate user on the same . These logs have durations from around one to ten hours each day. We truncate all logs to 90 minutes during which 18 users record their location. Synthetic mobility traces are generated using Levy Walk mobility model [73] with 20 (10 slow, 10 fast moving) and 40 (30 slow, 10 fast moving) nodes. Details about trace settings are provided in Table 4.2.

Table 4.2 Levy mobility trace used for simulation

Description	**slow**	**fast**
Exponent of step length distribution	1.6	
Exponent of pause time distribution	0.6	
Velocity of node	1m/s	5m/s
Minimum step length	5m	25m
Minimum pause duration	30s	7.5s
Maximum pause duration	600s	60s
Simulation area	500×500m	

Table 4.3 Simulation parameters

Description	Range [**default value**]
Number of services	[**20**]
Number of service providers	[**20**], 40
Repetition of each service	1, [**2**], 3, 4
Request timeout	10, [**15**], 20, 30
Request rate per node	0.2, [**0.4**], 0.67, 1/min
Trace duration	[**10 hours**]
Transmission range of device	[**100m**]

4.5.1 Simulation Setup

All plots show the average of five simulation runs and the error bars are plotted on the minimum and maximum values in those runs. Table 4.3 summarizes the simulation parameters. In all following plots, default values of these parameters are used unless it is mentioned otherwise under the plot. Parameters used for forwarding schemes are EBR-Encounter rate window=10 minutes, t_{av}=10min for TT and 0.5min for MT. A total of seven input/output types are used to create $C_2^7 = 21$ unique services. Service s_{17} is dropped to leave 20 unique services that are provided at 20 nodes in the network. Note that for a service repetition level of 3, all services are provided by 3 unique nodes (randomly selected) such that every node provides exactly 3 services.

Service requests are only generated for services with $k \geq 4$ and the average execution time of service is set to 30s. Each service request times out after 15 minutes so that previous service requests do not overload the system. Thus, services are considered incomplete when they are not complete within 15 minutes. In all simulation results, service requests are not generated in the last 15 minutes (duration of request timeout) of the trace duration. We consider a simulation analysis including both a transient and a steady regime phase. The transient regime lasts for approximately the initial 30 minutes, after which all nodes acquire information about network resources and system enters the steady regime. In the transient regime the system starts with no initial service loads. Therefore, delays of only those compositions are considered that are generated once the system is well inside the steady state regime (service requests that are generated after the first two hours of simulation, are considered).

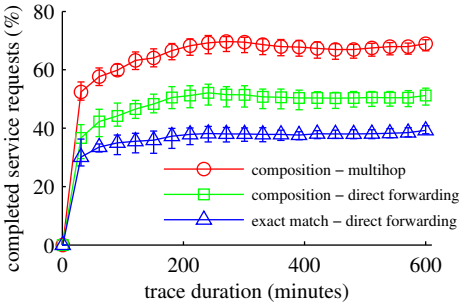

Figure 4.5 Higher service completion in multi-hop composition

4.5.2 Direct and Multi-hop Forwarding

Figure 4.5 shows the service completion rate in the Levy Walk mobility trace. In all these cases *local* level of awareness is used to construct service graph. The search for a single service that matches the request exactly only fulfills 40% of the requests.

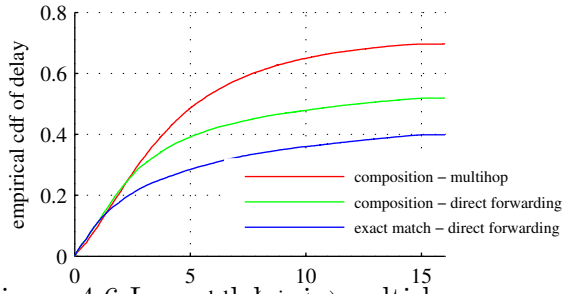

Figure 4.6 Lower delay in multi-hop composition

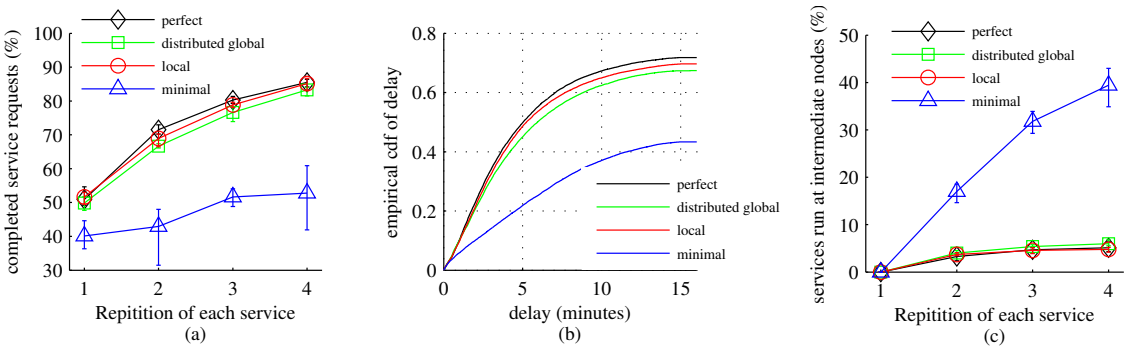

Figure 4.7 Levy Walk: With different levels of awareness (a) service completion, (b) delay, and (c) services run at intermediate nodes

However, when the condition is relaxed to compose multiple services to meet the requirements, about 50% of service requests are completed. As a further improvement, when multi-hop forwarding paths are used to forward requests and service outputs, 70% requests are completed. This shows the significant improvement that service composition with multi-hop forwarding paths can have over other mechanisms. While such mechanisms are discussed in [33, 86], our algorithm tolerates disconnections and utilizes paths that become available over time. As a result, in addition to utilizing services at devices with directly connected paths to a node, ones that are present in the nearby vicinity can be invoked at the expense of slightly higher delays. However, in opportunistic networks, multi-hop forwarding still yields lower delays as compared to direct one-hop forwarding. This is shown in Figure 4.6 where 50% of compositions are made within 5 minutes using multi-hop forwarding, whereas it takes 12 minutes

in direct forwarding. In case when only exact match is searched in the network, only 40% requests are completed after 15 minutes. The CDF plot (proportion of samples less than a value) is based on the aggregate delay per service from five simulation runs. As seen from Figure 4.5, the percentage of completed services increases sharply in the first 30 minutes (transient regime) and then becomes steady. Therefore, in Figure 4.6 and later, delays of only those compositions are considered that are generated once the system is well inside the steady state regime (after first two hours of simulation).

4.5.3 Levels of Awareness

While there is significant improvement in service completion rate and delays when some knowledge about temporal distances from other nodes is used, there is not much difference in the performances of perfect, globally aware and local schemes. This is clear from Figure 4.7a where in a range of service densities, completion rate of perfect, distributed global and local levels of awareness are within 3% of each other. Figure 4.7b shows the delays when each service is repeated twice i.e. two different nodes provide that service. The local scheme has delays between the globally aware and perfect scheme as shown in Figure 4.7b. One explanation for this behavior is that approximation for temporal distance between other nodes is better than delayed information of actual temporal distance between those nodes. Notice that at the time of request initiation, in perfect knowledge, a node knows exact distance between other nodes only at that instant of time - a node does not know the exact distance between other nodes in future when the first (or second, third etc.) service in composition will be complete. Thus, during the time a forwarded request is received on a device, the network topology changes so much that it does not make much difference if local knowledge was used or perfect knowledge to estimate distances between other nodes for selection of sevices/devices. Since, all three schemes have similar performance; we

select the scheme with local knowledge for further analysis as it is more light weight (requires exchanging $O(N)$ temporal distance and load values per contact instead of $O(N^2)$).

In Figure 4.7c, we validate the selection of correct devices to compose all services. At different service densities, percentage of services that are run at intermediate nodes while they were enroute to some other device is considered (i.e. when required service is opportunistically found at a device different from one selected by the algorithm). While this percentage is quite high around 40% when minimal knowledge about network is available, it is reduced to under 5% when some knowledge about network topology is used. This validates that our service selection algorithm selects the correct devices to execute services in a composition.

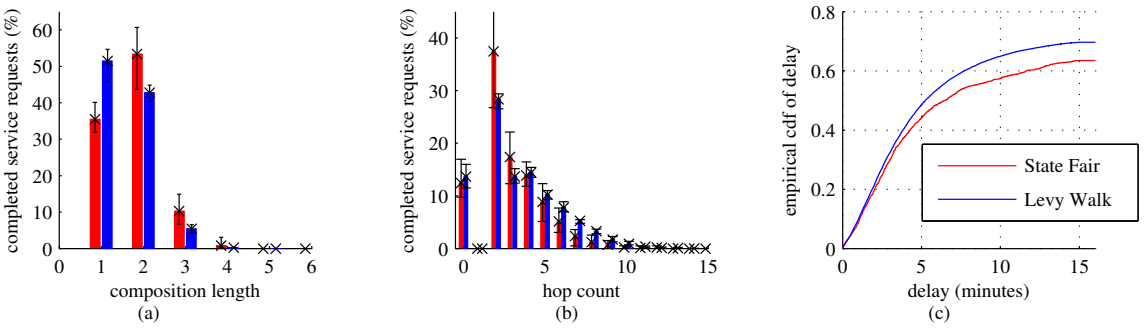

Figure 4.8 State Fair [red] and Levy Walk [blue] mobility trace: (a) composition length, (b) hop count, and (c) delay profiles for multi-hop service composition

4.5.4 Composition Length, Delay, and Hop Profiles

Figure 4.8 shows the detailed profile of multi-hop service composition for State Fair and Levy Walk mobility trace. Since, State Fair trace last only 90 minutes, we have performed extensive simulation on Levy walk with a duration of 10 hours to evaluate performance results in steady state. The results in this section are used to

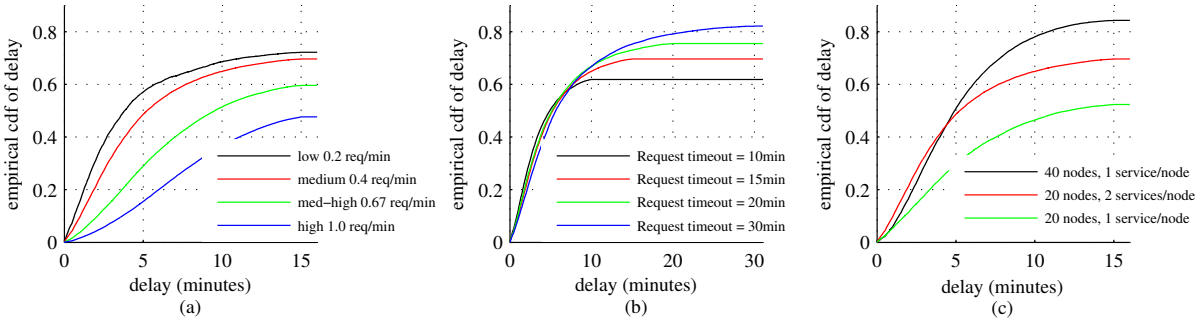

Figure 4.9 Levy Walk: Higher service completion and lower delay at (a) low loads, (b) long request timeout, and (c) high node, service density for multi-hop service composition

demonstrate that valuations based on Levy Walk traces are close to real mobility traces in environments similar to state fair.

Most compositions are comprised of 1 or 2 services and then there are some of three, and four services as shown in Figure 4.8a. In our analysis, services are distributed such that every service is provided by at least one node in the network. However, as described earlier, restricting the completion to the exact match degrades performance. Therefore, all compositions (around 60%) of length more than 1 that are shown in Figure 4.8a are optional, and are made to achieve higher service completion rates and minimal delays. However, these compositions come at the expense of a few extra hops as shown in Figure 4.8b. Most of the times (about 80%) complete services are composed by using less than 5 hops. Around 15% of requested services are found at the device itself. This causes a hop count of zero as shown in Figure 4.8b. If the service is not found at the requesting devices, service completion incurs at least 2 hops (one to send request and one to receive results). Figure 4.8c shows the empirical CDF of the time taken from service request to receiving composed results, routed back from the last node in the composition path. Around 60% of completed service requests have a delay of less than 10 minutes whereas close to 50% of these

are completed in less than 5 minutes. Each service is repeated twice in this run. For higher number of repetitions, delays are lower and percentage of completed services reaches close to 100% (85% for four repetitions, see Figure 4.7a). This means that for, reasonable service densities, applications that can tolerate delays to the order of 5 to 10 minutes, opportunistic networks can provide a reasonable service completion rate in an environment similar to State Fair.

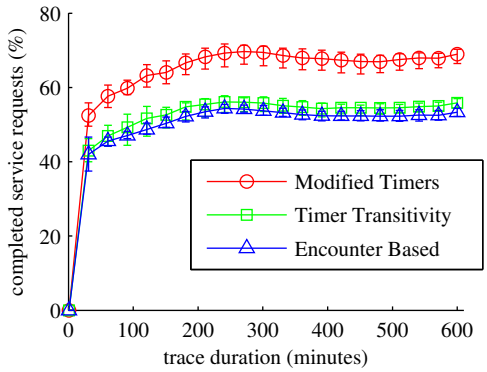

Figure 4.10 MT with better service completion

4.5.5 Effect of Forwarding Scheme, Request Load, Request Timeout, Node and Service Density

Figures 4.10 shows the service completion rate of 3 different underlying forwarding schemes used in the service selection algorithm. Only single copy of each request is forwarded. MT shows better delivery rate (about 30-40% higher than other schemes). Network performance is also analyzed under different service load conditions and levels of connectivity. Requests rates are varied from 0.2 to 1 request per minute per node. Figure 4.9a shows that service completion rate decreases under high loads. At high node/service density Figure 4.9c shows that almost 85% of all requests can be composed. Also, when a service request does not timeout for a longer time, more re-

quests can be completed [Figure 4.9b]. Thus, while service compositions can be made in a moderately connected network, our algorithm adapts well in sparse scenarios with higher service request loads and short request timeout duration. Note that variation in load greatly changes the delay profile under 5 min as shown in Figure 4.9a. This is because higher loads result in longer service queues and services take longer to execute even when a connected path to service provider exists. Whereas variation in request timeout changes the delay profile at over 10 min as shows in Figure 4.9b. This is because longer request timeouts keep the service requests in the queue which are otherwise dropped without completion. As a side effect, average service load at nodes increases slightly. For this reason, 40% of compositions are made in 4.5 min with request timeout of 30min but only in 3.5min with timeout of 10 min due to change in average load at nodes. Figure 4.9c shows that average delays are lower when 40 services are distributed in 40 nodes instead of 20 nodes even though the number of generated requests is double in a network with 40 nodes. This is because the network with 40 nodes is more dense and connected.

4.6 Characteristics of the Algorithm

The previous section showed performance results for service composition in a general scenario whereas in this section we aim to analyze key characteristics of the algorithm and the effect of specific features (while simplifying other details).

Thus, to compare performance under different mobility characteristics, we consider all nodes to be moving at the same speed (unlike the slow and fast moving nodes as reflected in Table 4.2) but vary the key distinctive feature of three different mobility models. Similarly, to analyze the effect of composition length (number of services to be composed to complete a request), we run simulations for requests made *only* for services of compositions of length one and compare against a separate sim-

ulation run for requests made *only* for services of composition length two and so on. Then, we characterize the effect of service distribution and load estimation on service composition algorithm. We also analyze accuracy of the algorithm by comparing the composition cost computed in service graph and the actual delay incurred to complete a service.

4.6.1 Mobility Characteristics

Fig. 4.4 shows performance of service composition under varying mobility parameters in Levy walk, SLAW and HCMM mobility models. These models provide a wide range of mobility characteristics to thoroughly evaluate performance of service composition: HCMM - movement in communities; SLAW - a clustered environment; and Levy walk - uniformly distributed mobile users. The key distinctive feature of SLAW is the Hurst Parameter that controls the degree of self-similarity and clustering in the environment. The flight length and pause time distribution are similar in Levy walk and SLAW so we do not vary those parameters. The key distinctive feature of HCMM is the rewiring probability which establishes the frequency of back and forth travel of nodes between communities. For all these mobility models we also study the effect of node density by keeping the number of nodes constant but varying the simulation area.

Fig. 4.4a shows that under similar user density, the number of completed services is greater in HCMM than SLAW, which in turn is higher than the number of completed services in Levy walk, e.g., for 20 users in simulation area size 700m (simulation area 700m×700m) 70% services are completed in HCMM but only 58% services are completed in SLAW and 35% in Levy walk. This is because nodes that stay closest to each other in communities under HCMM, are still closer in clusters

under SLAW as compared to uniform distribution in the whole simulation area under Levy walk.

Fig. 4.4a also shows that percentage of completed services decrease as user density decreases e.g. completed services in HCMM decrease from 81% to 64% when simulation area increases from 500m×500m to 900m×900m. A similar trend is followed for SLAW and Levy walk.

Fig. 4.4b shows the effect of Hurst Parameter. There is no significant trend at high node densities but the performance of service composition decreases at lower user density (20 users in 900m×900m area). From visual inspection of trajectory plots, Hurst parameter of 0.85 shows longer flights spread in larger area than Hurst parameter of 0.55. However, in several random generations of mobility plots we have found a great variation in distribution of waypoints for the same value of Hurst parameter. Also, since most real world traces have Hurst parameter value close to 0.75 [5, 53] we use this value for all the traces when performance under SLAW is compared with Levy walk or HCMM.

Fig. 4.4c shows the effect of rewiring probability in HCMM. Initially there is a sharp decrease and then an increase in completed services when rewiring probability is increased from zero. This is because, more services are completed when all nodes stay in the same community (rewiring probability = 0) versus when some of the nodes leave before completing the pending requests (rewiring probability = 0.1). However, as the back and forth travel of nodes between communities increases (rewiring probability > 0.4), there is an additional advantage of having services from a different community to be available. As a result, with increased value of rewiring probability, there is again an improvement in the percentage of completed services. However, the advantage of travel between communities also depends on how far the communities are. Thus, least number of services are completed at rewiring probability of 0.1 in

500m×500m area but in a larger area of 900m×900m the probability increases to 0.4. The advantage of traveling is manifested at higher rewiring probability (0.4 instead of 0.1) in 900m×900m because the communities are further apart and it takes more time to travel between communities. Thus, a small amount of travel to other communities decreases performance, but a lot of travelling between communities again improves performance of service composition. In all simulations, we use the value of rewiring probability to be 0.1 when comparing performance against Levy walk and SLAW.

4.6.2 Load Estimation

We explore performance of service composition algorithm with and without an estimate of load i.e. we consider

1. LA: load aware service composition (algorithm that takes into account the temporal distance as well as load at other nodes), and
2. NLA: load agnostic service composition (algorithm that takes into account the temporal distance and but does not take into account the load at other nodes)

Fig. 4.11 shows that load aware service composition (LA) improves performance by 10% in a more connected network (20 users in small area 500m×500m), but does not make much difference in sparsely connected networks (20 users in large area 900m×900m) in HCMM mobility. LA performs better than NLA in more connected networks, because delay due to sparse contacts is comparable to delay due to load at individual nodes. LA performs similar to NLA in sparse networks because delay due to sparse contacts is more dominant than delay due to load at individual nodes, and knowledge about load does not add much value in selecting which services to compose. Thus, the requirement to keep track of load at other nodes is critical when the forwarding delays are comparable to the delay due to queuing and execution of service requests.

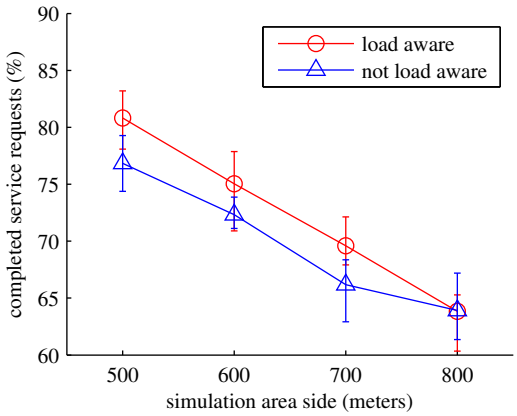

Figure 4.11 Load–aware service composition (LA) improves performance in a more connected network, but does not make much difference in sparsely connected networks

4.6.3 Estimated Cost and Actual Delay

In this section, we analyze the accuracy of values used to estimate the time required to complete a service as described in Section 4.3. Figure 4.12 shows that estimated cost for 70–80% of completed services is within 4 minutes of actual delay in HCMM, SLAW and Levy walk mobility models. Also, the difference between estimated cost and actual delay is less than 2 minutes for 40% of the completed services in all three models. This shows that the distributed *local* knowledge algorithm for service composition is fairly accurate in estimation and selection of optimal services that can be composed to complete the request.

Fig. 4.13 shows that estimated cost of incomplete services is 50% accurate (i.e. cost is greater than 15 minutes for 50% of incomplete services[2]) in Levy Walk but inaccurate in HCMM and SLAW mobility models. Notice, that this is not the overall accuracy, but the accuracy of *only* incomplete service requests which are a small percentage of the total number of service requests.

[2]Note that services are considered incomplete when they are not complete within 15 minutes. Thus, cost estimate of more than 15 minute is considered accurate for incomplete services

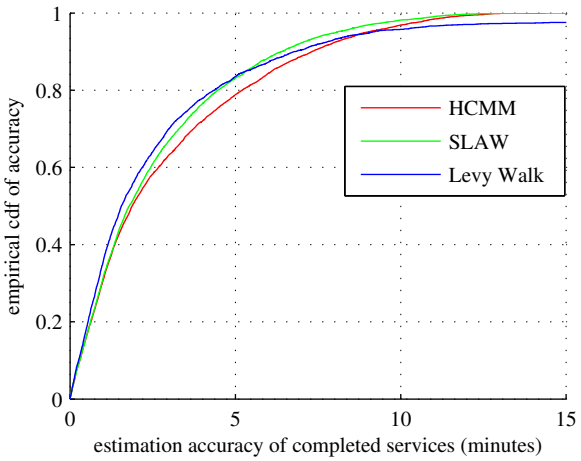

Figure 4.12 Estimated cost for 70–80% of completed services is within 4 minutes of actual delay in HCMM, SLAW and Levy walk mobility models.

This demonstrates that temporal distance is not a very good measure when the user movements contain structures of clusters and communities but has reasonable accuracy when user movement is more evenly spread in the surrounding area. The reason for inaccuracy in HCMM and SLAW is that nodes leave the community at random which results in large number of incomplete services. Using temporal distance alone does not predict which nodes may or may not leave the community/cluster in future under HCMM/SLAW. The possible solutions in such a case is to either introduce redundancy by requesting services with more than one node or to have a mechanism to predict user movement (i.e. which users may or may not leave the community) through other sources of information (user schedule, history of visits etc.).

4.6.4 Sensitivity to Composition Length and Service Distribution

The required number of services that are composed to complete a request greatly affects the performance of service composition. In addition, performance improves when services are distributed in the same pattern as the requests are gen-

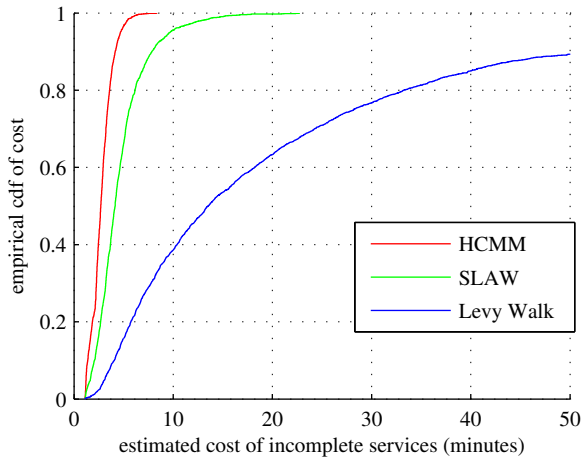

Figure 4.13 Estimated cost of incomplete services is 50% accurate (cost >15 minutes) in Levy Walk but inaccurate in HCMM and SLAW mobility models.

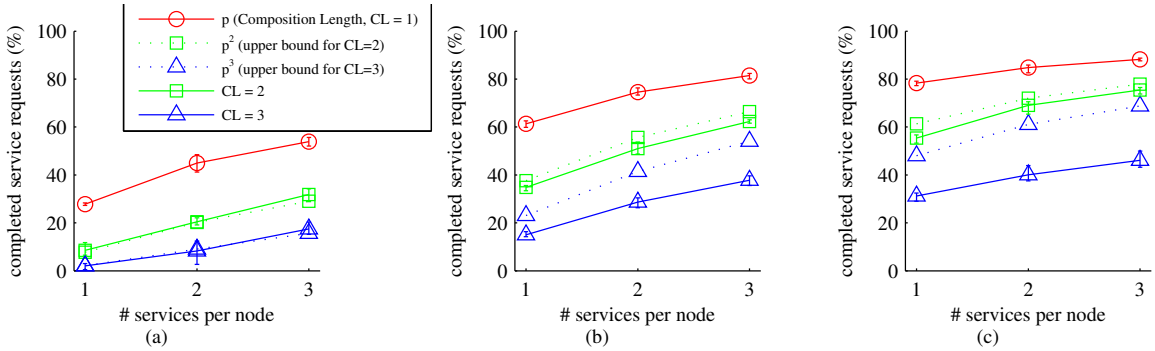

Figure 4.14 Sensitivity analysis for composition of 1,2 or 3 services in: (a) Levy Walk, (b) SLAW, and (c) HCMM mobility models.

erated (i.e. more commonly requested services are provided at more nodes than the rarely requested services). To analyze only the effect of composition length and service distribution, a slightly different set of services is used from the one described in Section 4.5.1. A total of 20 input/output types are used to create 20 unique services of functionality equal to one $k = 1$. Thus, the created services are $\{s_{1\,2}, s_{2\,3}, s_{3\,4}, ..., s_{8\,9}, s_{9\,10}, s_{10\,11}, ..., s_{19\,20}, s_{20\,1}\}$. Each of these services is provided by two nodes selected uniformly at random.

4.6.4.1 Composition Length

Fig. 4.14 shows sensitivity analysis for composition of 1,2 or 3 services in Levy walk, SLAW and HCMM mobility models for a range of service densities. The performance of service composition is analyzed with different request patterns as follows: Service requests require

1. exactly a single service – composition length = 1,
2. composition of two service components – composition length = 2,
3. composition of three service components – composition length = 3.

For example, a request with input 3 and output 5 requires two service components s_{34} and s_{45}. The simulation results show that the performance degrades when more service components are required to complete a service request. It should be noted, that this result actually elaborates the earlier deduction in Figure 4.5 where composition leads to better performance. The difference between these two results is in the simulation setting. Figure 4.5 shows the results for a general service distribution and request pattern [Section 4.5.1] - in this setup, allowing for composition of more than one service significantly improves the percentage of completed services. In contrast, Figure 4.14 shows results of a complete simulation run where services with a fixed composition length are requested. It shows the percentage of completed services by making requests that only require a single service. The impact of composition by making requests that require composition of two services in a separate simulation run is shown in Figure 4.14. More specifically, the percentage of completed services decreases when two or more services are composed *for every request*. This is to emphasize the point that composition by itself does *not* lead to better performance [Figure 4.14] but rather composition leads to better performance *only* when the cost of selected composition is compared and found better against the cost of all other

possible compositions [Figure 4.5] (as is done in our proposed service composition algorithm).

Figure 4.14 shows sensitivity to composition length in different mobility environments with a controlled pattern of service requests. Specifically, if the ratio of completed services is p (where $0 \leq p \leq 1$) when composition length = 1, then the ratio of completed services is less than p^2 when composition length = 2, and less than p^3 when composition length =3. This can be explained by assuming the probability of completing a single service to be p. Assume that probability to complete a service is independent of probability to complete other services in the request, the particular service that is requested and the node that requests it. Then probability of completing a request for composition of two services is p^2 and probability of completing a request for composition of three services is p^3. In simulation, the actual ratio of complete services is slightly under p^2 and p^3 because available time–per–service reduces when a request needs more than one services to be composed. Thus, probability of completing a request for composition of two services is slightly less than p^2 due to smaller available time–per–service.

The value of p^2 and p^3 serves as an upper bound (estimate) to the ratio of completed services for composition length = 2 and 3 respectively. This upper bound is close (slightly greater than actual ratio of completed services) under Levy Walk Fig. 4.14a, but is not close (a lot greater than actual ratio of completed services) in SLAW [Fig. 4.14b] and HCMM [Fig. 4.14c]. This is because nodes and services are located in clusters/communities in SLAW/HCMM. Unlike levy walk, the probability of completing a service is greatly affected by whether the requested service is found in the same cluster or community in SLAW/HCMM. When a single service is requested, most of the completed services are located within the same cluster/community in SLAW/HCMM. Therefore, the estimate of completing a single service is biased to-

wards assumption of availability of service in the same or nearby cluster/community. As a result, whenever one or more services required to complete the request are not available in the same cluster or community, the probability completing the request becomes way lower than the upper bound of p^2 and p^3 for the composition length 2 and 3 respectively.

4.6.4.2 Service Distribution

Figure 4.15 shows that performance of service composition improves if less popular (requested) services are replaced by more popular (requested) services. To analyze sensitivity to service distribution consider that half of the services (10 out of 20 services) are requested 3 times as often as the other half. Then following two scenarios are of interest:

- Uniform distribution: Each service (of all 20 services) is provided at two different nodes. The total number of services including repetition is 40 such that every node provides two services.
- Proportional distribution: Each of the 10 services that are requested more often is provided at three different nodes, and each of the 10 services that are requested less often is provided at only one node. The total number of services including repetition is still 40 such that every node provides two services.

Also, services and nodes are selected uniformly at random for each selection. Requests are made such that they require a composition of two services. Results show an improvement of close to 20% when services are distributed proportional to request rate as demonstrated in Figure 4.15 under Levy walk, SLAW and HCMM. This provides a key insight that replacement of less popular services with more popular services to match the request pattern increases percentage of completed services. The less popular service is replaced by a more popular service at a node selected

randomly from all nodes providing the less popular service. Performance can be improved even further by considering location and mobility characteristics (instead of random selection) of a node on which service is replaced.

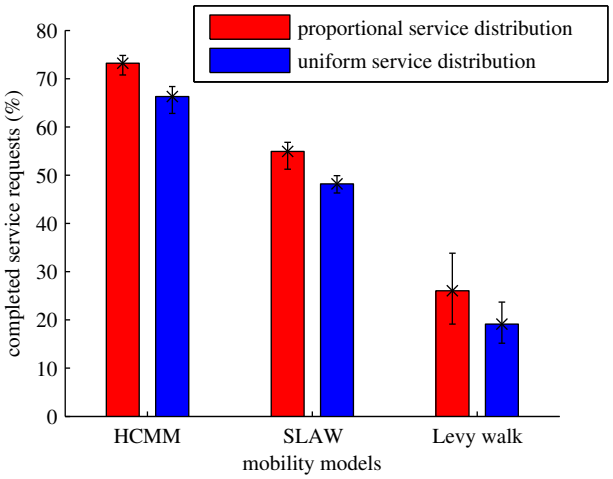

Figure 4.15 Performance of service composition improves if less popular services are replaced by more popular services.

4.7 Summary

In this chapter, we proposed a novel algorithm for service composition in opportunistic networks. With the proposed algorithm, mobile users can benefit from a larger set of services available locally in an environment. Proposed service composition makes efficient service selections from devices that are in close proximity. Key insight used in the solution is that temporal distance between devices provides a direct measure for reachability of nodes when an end-to-end connected path does not exist. This way, based on a service graph, a composition sequence can be selected to meet the requirements of service request while any routing scheme can be used to forward service parameters.

Through extensive simulations on real and synthetic mobility traces, we conclude that using only local knowledge about temporal distances and service loads, a large percentage of services can be composed in an opportunistic network using multi-hop paths among devices. In order to complete a request, searching for possible compositions is better than looking for a single service that matches the request exactly. Also, use of multi-hop forwarding paths is more efficient than one-hop direct forwarding. Performance of service composition improves (more requests are completed with lower delays) under high service density, high node density, clustered environments with same node density (like HCMM or SLAW in comparison with Levy walk), long request timeout duration and low request rate. In scenarios when some services are requested more often than others, performance improves when less popular (requested) services are replaced by more popular (requested) services.

Future work will explore the key insight about replacement of less popular services with more popular services - specifically based on location and mobility characteristics of the user providing a particular service.

CHAPTER 5

COLLUSION–RESISTANT INCENTIVE–COMPATIBLE ROUTING AND FORWARDING

In this chapter we present a new credit scheme called CRISP (Collusion-Resistant Incentive-ComPatible routing and forwarding) in which truthful behavior in routing and forwarding maximizes a node's profit in an opportunistic network.

5.1 Introduction

The basic CRISP mechanism requires nodes to pay credit to receive a data packet and later obtain credit for forwarding that packet. Nodes must also pay credit to send new data packets of their own. Intuitively, selfish nodes that fail to forward packets will not be able to send their own packets, and malicious nodes are prevented from flooding the network. Further, routing black holes are greatly limited due to the required payment for receiving the data. In CRISP, the required payment for receiving a packet is less than the reward for successful forwarding.

Although these mechanisms are intuitive, ensuring both collusion resistance and incentive compatibility for routing as well as forwarding is challenging. In this work, we describe a principled method for setting the payment parameters. In particular, we first transform a contact graph of the network into an instance of a network flow problem and model the loss of packets in the network as a queuing theory problem with *reneging*. With this model, we find the optimal operating point of individual nodes to maximize overall network flow. We then design our credit–based incentive scheme so as to maximize the profit of a node when it operates at this optimal

operating point. CRISP makes it the selfish choice to receive and forward data packets in the most efficient manner for the network as a whole. To make the scheme collusion–resistant, the reward for forwarding a packet is based on the routing utilities of relay nodes instead of a fixed amount and digital signatures are used to enforce specific conditions on routing utilities of relay nodes.

CRISP is thus the first credit scheme that is appropriate for opportunistic networks. It is fully distributed and requires neither a central bank nor *a priori* knowledge of routing paths. CRISP is robust to flooding and blackhole attacks and provides incentives for nodes to participate fully in routing and forwarding operations.

In addition, we also present the first consistent and secure distributed payment framework (DPF) for maintaining and tracking the credits in a distributed manner without any centralized trusted authority. In particular, payments are secured using a Merkle hash tree [94] over the transaction history, and the histories are monitored by other nodes in the network.

5.1.1 Organization

We now layout the organization of the rest of the chapter. In Section 5.2, we provide background on opportunistic networks and credit schemes so as to best motivate our design choices. Section 5.3 describes a model of the opportunistic network contact graph as a flow network and shows how we further model each node as a M/M/1 queue with reneging. Based on this model, Section 5.4 shows how to find the optimal arrival rate under a given function of loss (the *attenuation*). We then describe our credit scheme in Section 5.5, with payments and rewards set so as to make nodes obtain their maximum profit when accepting and forwarding data at the optimal arrival rate. Section 5.6 describes the distributed payment framework (DPF) to make and receive payments in an opportunistic network. Section 5.7 shows the

results of analysis and simulations that demonstrate the effectiveness of our approach. Finally, we provide summary of the chapter in Section 5.8.

5.2 Design Considerations

In an opportunistic network, a node acts as a: *source* – generates data; *relay* – receives, stores and forwards data; and *destination* – receives data. Therefore, the possible cheating actions in each role can be: source – generates dummy data (flooding attack); relay – drops data (black–hole attack)/ randomly forwards data (to earn credit); or destination – does not acknowledge received data. In addition, nodes can also collude with each other for such attacks. Payment for generating data and a reward for acknowledging received data are used in [91] to secure against cheating by a source and destination. However, no existing credit schemes or security frameworks secure the behavior of a relay in face of collusion (we discuss further in Section 2.3). To this end, CRISP ensures truthful forwarding and honest reporting of routing utility even when nodes collude. We now describe the key concepts used to design CRISP.

5.2.1 Routing and Forwarding

In an opportunistic network, nodes come into contact with each other at different times and locations. Upon contact with another node, a data packet is forwarded only if the encountered node is *more* likely to deliver the packet. This likelihood of delivery is computed by a routing/forwarding scheme that assigns every node a utility value for each destination where a node with better likelihood of delivery has a higher utility. For example, bubble rap [46], maxprop [14], and others [59, 83] (including our work on data forwarding [Chapter 3]) assign utility values for social based forwarding, vehicle based routing, and general opportunistic networks, respectively. Let U_v^x

represent the utility value of node v for destination node x that is computed by a corresponding routing scheme (e.g., [14, 46, 59, 83]). A packet for destination node x is forwarded from node v to another node w if node w is more likely to deliver the packet, i.e., if $U_w^x > U_v^x$. A packet also has an expiration time such that if a packet is not delivered by that time, it is dropped/lost at the node. In this chapter, we consider the problem of truthful *reporting* of utility, with the assumption that the node *knows* its true utility (e.g., a node only updates its utility based on contact with a (trusted) node with high reputation value).

5.2.2 Incentive Compatibility

Incentive compatibility means that a node earns highest reward when it behaves honestly. There are two related concepts: strong incentive compatibility – honest behavior is the best strategy irrespective of the strategy of other nodes; and weak incentive compatibility – honest behavior is the best strategy with the requirement that other nodes behave honestly. It was proved by Zhong et al. that forwarding cannot be incentive compatible in the strong sense[1] [92]. Thus, we use the term incentive compatibility to mean weak incentive compatibility. Typically, credit schemes are incentive compatible in forwarding, i.e., a relay earns highest reward when it forwards the data to another relay or the destination itself. However, none of the existing credit schemes for opportunistic networks is incentive–compatible in routing.

The distinction between the forwarding and the routing stages was first highlighted by Zhong et al. for ad hoc networks [92]. Routing stage comprises of computing correct value of routing utility and forwarding stage comprises of sending the packet to the right relay based on the protocol. Techniques from mechanism de-

[1]strong incentive compatibility is equivalent to a mechanism in which honest behavior is the dominant strategy

sign [65], are adapted by Zhong et al. to make the routing *and* the forwarding stage incentive–compatible [92]. However, these techniques are not applicable in opportunistic networks because the complete path from source to destination is not known. A related problem is that of opportunistic routing in mesh networks, as the path is determined on a per–hop basis. Wu et al. make the routing incentive–compatible by a carefully designed incentive scheme [89]. However, the forwarding stage is not considered. Then, Chen et al. make forwarding in mesh networks incentive compatible with the *assumption* that routing is incentive compatible [27]. Thus, a relay makes maximum profit by honestly behaving in forwarding *only if* it is assumed that the relay behaves honestly in the routing stage. When considered together, however, routing and forwarding are not incentive compatible because a node can make a higher profit by cheating in both routing *and* forwarding. Therefore, it is important to consider routing and forwarding together, without assuming that nodes behave honestly in either stage.

5.2.3 Collusion

In ad hoc networks, nodes declare a cost of transfer, and it has been shown by Wang et al. that no mechanism can be incentive compatible when nodes collude in declaring the cost [88]. To bypass this restriction, we fix the cost of forwarding, i.e. every node gets the same minimum reward upon forwarding. Since every relay that takes part in the forwarding gets a reward, colluding nodes can add fake relays in the path. As a result, even those relays that do not take part in the forwarding get a reward. One recent scheme, Mobicent [26], addresses this issue by decreasing the amount of reward based on the number of relays in forwarding path. This approach, however, requires a large percentage of payment to the central credit authority (e.g. 57%, 73% in case of 3, 4 relays). It is not clear how to redistribute this over payment

back to the nodes such that the original scheme remains incentive compatible. For example, redistribution of extra payment among relays that contribute allows nodes to increase reward by adding fake relays. In addition, Mobicent does not secure the routing behavior of nodes, as a relay can increase its reward and decrease network throughput by modifying its routing utility to receive and randomly forward those packets.

5.2.4 Design Principles

CRISP is designed based on following principles:

1. A relay forwards data to a node with higher routing utility than a specific value;
2. Reward to a relay is based upon successful delivery of data, and the amount of reward is proportional to the improvement in utility made by the relay;and
3. The ratio of payment (for receiving data) to reward (upon forwarding data) is adjusted so that reporting incorrect routing utility results in reduced profit.

Principle (1) is used to secure against random forwarding. To enforce this requirement, a cryptographic technique of layered coins [60, 94] is used. Upon exchanging a packet, relays in contact sign (using their private key) the routing utility at which the packet is received (see [60, 94] for details). Suppose node v receives a packet destined for node x when its utility U_v^x is 0.2, then node v is allowed to forward the packet to another node w only if $U_w^x > U_v^x$ (in this case $U_w^x > 0.2$). Note that $U_v^x = 0.2$ is stored in the layered coin in an un-forgeable way, thus a relay cannot change this value. This prevents a node from randomly forwarding a packet.

Principle (2) is used to secure against collusions that add fake relays. Thus, if a node v receives a packet at utility U_v^x and forwards it to a node w with utility U_w^x, then the reward is proportional to $U_w^x - U_v^x$. As a node forwards more packets, its expected reward also increases. However, adding fake relays in the path does not

increase the total profit because U_w^x and U_v^x stay the same (U_w^x, U_v^x are stored in the layered coin).

Principle (3) is used to secure against collusions to modify declared routing utility. Suppose a node declares a routing utility $\hat{U}_v^x$ instead of its true utility U_v^x. If $\hat{U}_v^x > U_v^x$, the node receives more packets but it becomes less likely to forward (due to requirement of finding neighboring nodes with a $U_w^x > \hat{U}_v^x$) the packets. If $\hat{U}_v^x < U_v^x$, the node receives less packets, but is more likely to forward the packets. If $\hat{U}_v^x = U_v^x$, the node receives packets at a certain rate. This is called the optimal arrival rate as it maximizes the throughput (the number of packets that are delivered to destination) of the network. In our analysis, we formulate this scenario as a network flow problem. In CRSIP, a node imperatively forwards the packet as its reward on forwarding is greater than the payment made upon receiving the packet.

5.2.5 Network Flow and Optimization

Two important extensions of network flow are: i) generalized flow – a flow that is not conserved across an arc and ii) dynamic flow – a flow that takes time to travel across an arc. An opportunistic network is better modeled by a generalized dynamic flow, with multiple commodities and concave gains. The multiple commodities represent packets routed to different destinations and the concave gains can be used to model the packets lost/expired before delivery to the destination. A closely related problem of generalized flow with concave gains has been studied in [78, 85], and more recently Gross and Skutella prove hardness results for generalized dynamic flow [37]. To keep our analysis tractable, we drop the requirement of dynamic flow and model the opportunistic network as a generalized flow with multiple commodities and concave gains. Analytical results are validated through simulations.

5.3 System Model

This section describes: (i) the contact graph of opportunistic network; (ii) the flow network model for data transfer between nodes; and (iii) the queue model for data loss at nodes (data is dropped from the queue when not serviced in time).

5.3.1 Contact Graph

In an opportunistic network, nodes come into contact with each other at different times and locations. Consider the network operation in a time interval R of duration T (to the order of few minutes). We construct a directed contact graph $G = (V, E)$, where each vertex $v, w \in V$ represents a node in an opportunistic network with $n = |V|$, and each arc $(v, w) \in E$ represents a contact between node v and node w at any point in time during the interval R. Every node $v \in V$ in a network acts as a source, relay, and destination. Let d_{vw} represent the amount of data that can be sent from node v to node w during any number of contacts in the interval R. Note that d_{vw} is the duration of contacts multiplied by the bandwidth available for communication. We consider the network to be symmetric, i.e., if $(v, w) \in E$ then $(w, v) \in E$ and $d_{vw} = d_{wv}$. Note that $d_{vw} = 0$ if $vw \notin E$, and $d_{vv} := 0$.

5.3.2 Flow Network

Graph G can be modeled as a multi–commodity flow network G_g. In this network, the *amount of flow* is used to represent the *rate of data transfer*. Thus, the terms arrival, loss and forwarding rate of packets are equivalent to the incoming, lost and outgoing flow respectively. These terms are used interchangeably in rest of the chapter.

Let c_{vw} represent the flow capacity between nodes v and w. Then $c_{vw} := d_{vw}/T$, i.e., the constant *rate* of data transfer possible between node v and node w

for the entire interval R. Network G_g has multiple commodities of flows $x \in V$ which correspond to data en–route for different destination nodes $x \in V$. Let g_{vw}^x be a flow of commodity x. The amount of flow g_{vw}^x is used to represent the rate of transfer of data (destined to node x) from node v to node w. Data of a particular commodity is only transferred in one direction, i.e., if $g_{vw}^x > 0$ then $g_{wv}^x = 0$. Let b_v^x be the amount of flow of commodity x generated at node v. Flow b_v^x is used to represent rate for the data generated during the interval R or available at the start of interval R at node v. All flows are non–negative and the capacity constraint requires that the sum of flows of all commodities between nodes v and w is less than or equal to the capacity c_{vw}[2], i.e.,

$$\sum_{x \in V} g_{vw}^x + \sum_{x \in V} g_{wv}^x \leq c_{vw} \quad \forall v, w \in V, \tag{5.1}$$

Data packets have an expiration time, and a packet is lost at a node if it is not delivered before the expiration time. Let Q_v^x be the probability that a packet of commodity x is lost at node v. Then, if a node v receives packets for destination node x at the rate $\lambda_v^x = \sum_{w \in V} g_{wv}^x$, it is able to forward at the rate $\Gamma_v^x(\lambda_v^x) := \lambda_v^x(1 - Q_v^x)$ where $\Gamma_v^x(\lambda_v^x)$ is called the attenuation function.

Note that the data is lost at the node itself. This can be modeled in terms of a generalized flow network by splitting each vertex into two vertices joined by a new arc of flow λ_v^x, and then adding the attenuation function $\Gamma_v^x(\lambda_v^x)$ over the flow of this new arc. This process is similar to the conversion of a flow network with vertex capacities to a flow network with arc capacities. For notational simplicity, however, we avoid the standard notation and describe the flow conservation as follows. For each node and

[2]Note that $c_{vw} = c_{wv}$.

commodity, the sum of outgoing flows is equal to the sum of locally generated flows and the attenuated incoming flow, i.e., for all $v, x \in V \quad \sum_{w \in V} g_{vw}^x = \Gamma_v^x(\lambda_v^x) + b_v^x$.

5.3.2.1 Notation

Let $\lambda_{\mathbf{v}} = (\lambda_v^x, \forall x \in V - \{v\})$ represent the column vector at node v of the arrival rates (incoming flows) of all commodities except commodity destined to node v such that $\lambda_{\mathbf{v}} \in \mathbb{R}^{(n-1)\times 1}$. Let $\lambda = (\lambda_{\mathbf{v}}, \forall v \in V)$ represent the column vector of arrival rates (incoming flows) of all nodes and commodities except commodities destined to each node such that $\lambda \in \mathbb{R}^{n(n-1)\times 1}$. Let $\hat{\lambda}_{\mathbf{v}}$ and $\hat{\lambda}$ represent the corresponding optimal arrival rate vectors. Let $\mathbf{h}_{\mathbf{v}} = \lambda_{\mathbf{v}} - \hat{\lambda}_{\mathbf{v}}$ and $\mathbf{h} = \lambda - \hat{\lambda}$ represent the difference between a general and optimal rate vector where $h_v^x = \lambda_v^x - \hat{\lambda}_v^x$ is used to represent an individual component. Vectors $\mathbf{S_v}$ and $\mathbf{S}$ are used to show the sum of vector components, where $\mathbf{S_v} = (1, \forall x \in V - \{v\})$ and $\mathbf{S} = (1, \forall v \in V, x \in V - \{v\})$ are row vectors of ones. For example, $\mathbf{S_v}\lambda_{\mathbf{v}} = \sum_{x \in V-\{v\}} \lambda_v^x$ is the total incoming flow (excluding commodity v) at node v, similarly, $\mathbf{S}\lambda = \sum_{v \in V} \sum_{x \in V-\{v\}} \lambda_v^x$ is the total incoming flow at all nodes (excluding commodities destined to each node). The total outgoing flow (excluding locally generated flow) at all nodes is represented by $\Gamma(\lambda) := \sum_{v \in V} \Gamma_v(\lambda_{\mathbf{v}})$ where $\Gamma_v(\lambda_{\mathbf{v}}) := \sum_{x \in V-\{v\}} \Gamma_v^x(\lambda_v^x)$ is the outgoing flow at node v.

5.3.3 Queue Model

Packets received by a node have some expiration time. Thus, not all packets are forwarded. If a node v receives commodity x at rate λ_v^x, then the rate *available* to forward the received commodity is μ_v^x, such that

$$\lambda_v^x + \mu_v^x = c_v^x. \tag{5.2}$$

The net capacity of relaying commodity x through node v is limited by capacity c_v^x, where c_v^x depends on the capacity c_{vw} available with each node w, the rate b_v^x consumed by forwarding the data generated at node v, and the schedule for splitting available capacity c_{vw} for each commodity x. In our solution, we look at the properties of the optimal arrival rate λ_v^x for a given capacity c_v^x for each node and commodity.

The rate available for forwarding μ_v^x is different from the rate at which commodity x is actually forwarded $\Gamma_v^x(\lambda_v^x)$. For example, a node can receive data at the rate (in packets/min) $\lambda_v^x = 5$ with the available rate of forwarding $\mu_v^x = 10$, but only able to forward at $\Gamma_v^x(\lambda_v^x) = 3$. $\Gamma_v^x(\lambda_v^x) \leq \lambda_v^x$ because a node cannot forward more packets than it receives, and some packets expire when there is no contact opportunity to forward the packet.

To derive the packet loss rate, as an approximation, the arrival rate λ_v^x and available forwarding rate μ_v^x are modeled as a Poisson Process. This scenario represents a $M/M/1$ queue with reneging [10]. The arrival process is Poisson with rate λ_v^x and the service process corresponds to forwarding process which is Poisson with rate μ_v^x. Each node v is a server for a flow commodity x. The holding time of a packet in a queue is the time after which a packet is dropped if its service has not been completed, i.e. if the packet has not been forwarded. We assume a constant holding time δ_v^x/c_v^x, where δ_v^x is a dimensionless constant that gives the order of holding time with respect to the capacity of a node c_v^x.

5.4 Analysis

Using the system model from previous section we now derive the optimal packet arrival rates. The value of utility that a node declares determines the arrival rate of packets and consequently the packet loss rate. In this section, we derive the relation for packet loss and use it to find the optimal packet arrival rates that maximize

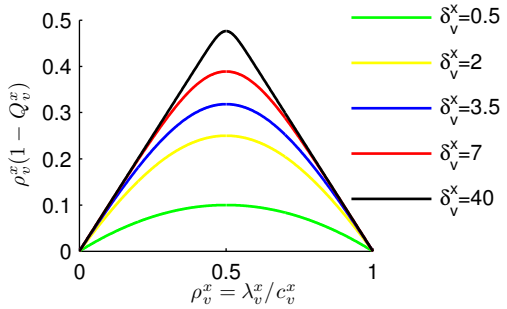

Figure 5.1 Forwarding rate is a concave function of packet arrival rate.

the network flow. We call it the optimal packet arrival rate, as it corresponds to the optimal value of utility that a node should declare to maximize the network throughput.

5.4.1 Attenuation due to Expired Packets

We use the result (equation (29) in [10]) for the probability of a dropped packet Q_v^x in a $M/M/1$ queue with reneging. Substituting μ_v^x from Eqn. (5.2) and replacing λ_v^x/c_v^x with ρ_v^x, the probability of packet loss is given by

$$Q_v^x = \frac{(1-2\rho_v^x)e^{\delta_v^x(2\rho_v^x-1)}}{1-\rho_v^x(1+e^{\delta_v^x(2\rho_v^x-1)})} \qquad \text{when} \quad \rho_v^x \neq 0.5,$$

where $Q_v^x = 2/(2+\delta_v^x)$ when $\rho_v^x = 0.5$. Though we were unable to prove concavity, through simulation studies, we find that for a fixed value of δ_v^x, the function $\rho_v^x(1-Q_v^x)$ is concave (first derivative decreases monotonically) in terms of ρ_v^x. Figure 5.1 shows the function $\rho_v^x(1-Q_v^x)$ for different values of δ_v^x. Using the known property, that perspective of a concave function is also concave (replacing ρ_v^x by λ_v^x/c_v^x and multiplying by c_v^x) [12], we find that $\Gamma_v^x(\lambda_v^x) = \lambda_v^x(1-Q_v^x)$ is concave in terms of λ_v^x. This result simplifies the optimization problem for flow maximization that is described next.

5.4.2 Optimal Flow with Concave Attenuation

In order to maximize the total flow delivered to the destination nodes, we minimize the total flow lost at individual relays given that all generated flow is pushed into the network. Using notation defined in Section 5.3.2.1, the total flow lost at all nodes is the sum of total flow lost at individual nodes for all commodities and is given by, $\mathbf{S}\lambda - \Gamma(\lambda)$.

Let the optimal incoming flow (arrival rate) at individual nodes be $\hat{\lambda}$ and the sum of optimal incoming flows be $L := \mathbf{S}\hat{\lambda}$. To find the properties of the optimal incoming flow that maximize the delivered flow, our objective is to minimize $L - \Gamma(\lambda)$ (total flow lost at relays) which is equivalent to maximizing $\Gamma(\lambda)$ such that $\mathbf{S}\lambda = L$. Also, flows cannot be negative, i.e., we have $\lambda \geq 0$, where $\geq$ is used for component wise inequality. Formally, this is described as,

$$\begin{aligned} \text{maximize} \quad & \Gamma(\lambda) \\ \text{subject to} \quad & \mathbf{S}\lambda = L, \qquad (5.3) \\ & \lambda \geq 0. \qquad (5.4) \end{aligned}$$

There is one additional capacity constraint on λ that we do not consider here. This is described in the form of individual flows g_{vw}^{x} in (5.1). One can solve the above optimization problem for the exact feasibility constraints. This problem is similar in structure to the problem of network utility maximization [51]. The concave attenuation function $\Gamma_{v}^{x}(\cdot)$ points to the law of diminishing returns, i.e., the probability of successfully forwarding a packet decreases as the node declares a high utility value in routing to receive more packets. However, we relax the flow capacity constraint (5.1) to constraint (5.3) in order to analyze the optimal rate $\hat{\lambda}$. The reason for this is that the topology in an opportunistic network is not static, and as nodes move around, the rate $\hat{\lambda}_{v}^{x}$ of one node (commodity) can be shifted to another node (commodity). In

addition, our simulations in a range of environments shows that honest behavior of nodes in routing and forwarding yields arrival rates that are very close to the optimal rate $\hat{\lambda}$ with constraints (5.3) and (5.4) only.

Based on our analysis of attenuation in Section 5.4.1, we assume that $\Gamma_v^x(\lambda_v^x)$ is concave and differentiable. Then, $-\Gamma_v^x(\lambda_v^x)$ is convex and $-\Gamma(\lambda)$ is convex because it is an increasing convex function of a convex function. Also, constraints (5.3) and (5.4) are affine and convex. Solving the Lagranian, the optimal point is the global maximum and Karush–Kuhn–Tucker conditions for optimality [12] give us

$$\nabla(-\Gamma(\hat{\lambda})) + \xi\nabla(\mathbf{S}\hat{\lambda} - L) + \sum_{v\in V}\sum_{x\in V-\{v\}} \xi_v^x \nabla(-\hat{\lambda}_v^x) = 0,$$

$$\forall v \in V, x \in V - \{v\} \quad \frac{\partial \Gamma_v^x(\hat{\lambda}_v^x)}{\partial \lambda_v^x} = \xi - \xi_v^x, \tag{5.5}$$

$$\forall v \in V, x \in V - \{v\} \qquad \xi_v^x \ \hat{\lambda}_v^x = 0, \tag{5.6}$$

where the Lagrange multipliers $\xi, \xi_v^x \geq 0$. Here $\xi \geq 0$ due to optimal value of L. It can be shown that $\Gamma(\lambda)$ for $\xi = 0$ is always greater than the case when $\xi < 0$ by changing constraint (5.3) to be $\mathbf{S}\lambda \leq L$. Equation (5.5) is the stationary condition, whereas (5.6) is the complementary slackness condition. These are in addition to the primal feasibility constraints (5.3) and (5.4). These optimality conditions can be further simplified. From (5.6), if $\hat{\lambda}_v^x > 0$, then $\xi_v^x = 0$ and $\partial\Gamma_v^x(\hat{\lambda}_v^x)/\partial\lambda_v^x = \xi$. So, our final result summarizes the optimality conditions of (5.3), (5.4), (5.5), and (5.6) as follows.

5.4.2.1 Main Result

Let $\hat{\lambda}$ be the arrival rate that satisfies primal feasibility constraints (5.3) and (5.4). Suppose there exists a constant $\xi \geq 0$ such that $\forall \hat{\lambda}_v^x > 0$,

$$\frac{\partial \Gamma_v^x(\hat{\lambda}_v^x)}{\partial \lambda_v^x} = \xi, \tag{5.7}$$

then $\hat{\lambda}$ is the optimal arrival rate, i.e., it maximizes $\Gamma(\hat{\lambda})$. Intuitively, this means that shifting flow from one path to any another does not increase delivered flow – in the next section we generalize this result.

5.4.3 Optimal Flow with General Attenuation

In the previous section, we derived the optimal arrival rate for a continuous, differentiable, and concave attenuation function. Our result on concavity of the attenuation function is from an approximate $M/M/1$ queue model. Therefore, in this section, we consider the general function $\Gamma_v^x : \mathbb{R}_+ \to \mathbb{R}_+$, without any requirement of continuity, differentiability, or concavity. The following derivations use the insight about optimality conditions from the case of the continuous concave function.

Theorem 1. *Let $\hat{\lambda}$ be the arrival rate that satisfies feasibility constraints (5.3) and (5.4). Suppose there exists a constant $\xi \geq 0$ such that $\forall \hat{\lambda}_v^x > 0$,*

$$\frac{\Gamma_v^x(\lambda_v^x) - \Gamma_v^x(\hat{\lambda}_v^x)}{\lambda_v^x - \hat{\lambda}_v^x} \geq \xi, \quad \forall \lambda_v^x \in [0, \hat{\lambda}_v^x), \text{ and}$$
$$\frac{\Gamma_v^x(\lambda_v^x) - \Gamma_v^x(\hat{\lambda}_v^x)}{\lambda_v^x - \hat{\lambda}_v^x} < \xi, \quad \forall \lambda_v^x > \hat{\lambda}_v^x.$$

Then $\hat{\lambda}$ is the optimal arrival rate, i.e. it maximizes $\Gamma(\hat{\lambda})$.

Proof. In order to prove optimality, we show that $\Gamma(\hat{\lambda}) \geq \Gamma(\lambda)$ for any λ that satisfies constraints (5.3) and (5.4). We can express any general rate $\lambda_v^x \geq 0$ in terms of the optimal $\hat{\lambda}_v^x$ as $\lambda_v^x = \hat{\lambda}_v^x + h_v^x$. Then the conditions in the theorem can be re–written as

$$\Gamma_v^x(\hat{\lambda}_v^x) - \Gamma_v^x(\hat{\lambda}_v^x + h_v^x) \geq -\xi h_v^x, \quad \forall h_v^x \in [-\hat{\lambda}_v^x, 0),$$
$$\Gamma_v^x(\hat{\lambda}_v^x) - \Gamma_v^x(\hat{\lambda}_v^x + h_v^x) > -\xi h_v^x, \quad \forall h_v^x > 0.$$

Summing the above two relations over all nodes and commodities, we get

$$\Gamma(\hat{\lambda}) - \Gamma(\lambda) \geq -\xi \mathbf{Sh}. \tag{5.8}$$

From constraint (5.3), $\mathbf{S}\lambda = \mathbf{S}\hat{\lambda}$. Substituting $\lambda = \hat{\lambda} + \mathbf{h}$,
$\Rightarrow \mathbf{S}(\hat{\lambda} + h) = \mathbf{S}\hat{\lambda} \Rightarrow \mathbf{Sh} = 0$. Substituting this result back into (5.8), we get $\Gamma(\hat{\lambda}) - \Gamma(\lambda) \geq 0 \Rightarrow \Gamma(\hat{\lambda}) \geq \Gamma(\lambda)$. □

5.4.3.1 Properties of Optimal Flow

The intuitive idea of optimal flow is simple. If *additional* packets are to be received at rate 10, the relay responsible for forwarding should be the one that can forward at rate 7 instead of the one that can only forward at rate 6. Therefore, with optimal allocation of flows, shifting individual flows from one node to another does not increase the overall forwarding rate. To see this property, let $Y_v^x(\phi) = \frac{\Gamma_v^x(\lambda_v^x + \phi) - \Gamma_v^x(\lambda_v^x)}{\phi}$. Then $\phi Y_v^x(\phi)$ is the increase in the forwarding rate for increasing the optimal arrival rate by ϕ. Similarly $\phi Y_w^x(-\phi)$ is the decrease in the forwarding rate for decreasing the optimal arrival rate by ϕ. The optimal allocation of flows requires that the increase in forwarding rate $\phi Y_v^x(\phi)$ at one node is always less than the decrease in forwarding rate $\phi Y_w^x(-\phi)$ at another node, for the same amount of change ϕ in arrival rate, i.e., $\phi Y_w^x(-\phi) > \phi Y_v^x(\phi) \Rightarrow Y_w^x(-\phi) > Y_v^x(\phi)$. This condition is true because Theorem 1 requires $Y_w^x(-\phi) \geq \xi$ and $Y_v^x(\phi) < \xi$ at all values of $\phi \in (0, \hat{\lambda}_w^x]$ and $\forall v \in V, x \in V - v$.

Table 5.1 CRISP operates at forwarding ratio 0.86 for any value of $\xi \in (0.05, 0.85)$.

Forwarding ratio	0.9	**0.86**	0.61	0.56	0.50
Forwarding rate Γ_v^x	9	**60**	61	62	60
Arrival rate λ_v^x	10	**70**	100	110	120

Also, the ratio of forwarding rate to the arrival rate is always greater than or equal to ξ (substituting $h_v^x = -\hat{\lambda}_v^x$ in Theorem 1 gives $\Gamma_v^x(\hat{\lambda}_v^x)/\hat{\lambda}_v^x \geq \xi$).

Consider the arrival (and corresponding forwarding) rates of a node in Table 5.1. Based on Theorem 1, for any value of $\xi \in (0.05, 0.85)$, the optimal arrival rate is $\lambda_v^x = 70$ with forwarding ratio 0.86. In a traditional credit scheme that rewards nodes based on the number of packets forwarded, a node maximizes its profit by maximizing its forwarding rate ($\Gamma_x^v = 62$) at a much lower forwarding ratio of 0.56. This is the key difference between CRISP and existing schemes and results in significant improvement in network throughput. We call ξ the threshold constant and evaluate its effects further in Section 5.7.

5.5 The CRISP Credit Scheme

In the previous section, we derived the optimal rate at which nodes should receive and forward the data. In this section, we present the design of a credit scheme in which receiving and forwarding at the exact same rate maximizes a node's profit.

5.5.1 Payments and Rewards

We require a node to make a payment $\omega > 0$ to receive a data packet. Nodes can then earn a reward $\kappa > \omega$ for forwarding a data packet. The rate of profit P_v^x a node v makes on receiving commodity x at rate λ_v^x and forwarding at rate $\Gamma_v^x(\lambda_v^x)$ is

given by $P_v^x = \kappa\Gamma_v^x(\lambda_v^x) - \omega\lambda_v^x$. Using notation from Section 5.3.2.1, the net rate of profit P_v at node v is given by $P_v = \sum_{x \in V-\{x\}} P_v^x = \kappa\Gamma_v(\lambda_{\mathbf{v}}) - \omega\mathbf{S_v}\lambda_{\mathbf{v}}$.

5.5.2 Credit Scheme

For $\omega > 0$ and $\kappa := \omega/\xi$, the exact payments to network and rewards from network for a node's role as source, relay and destination are as follows.

Relay: The payment to receive a packet is ω and (expected) reward upon forwarding a packet is $\kappa := \omega/\xi$.

Destination: The payment to receive a packet is ω and reward upon acknowledging the received packet is ω.

Source: The payment to generate a packet is a fixed amount to cover net cost on network.

5.5.3 Incentive Compatibility

Based on the above payment mechanism, we show that setting the correct ratio of payment to the reward results in maximizing the profit at the optimal rate of arrival. Hence, the credit scheme is incentive compatible.

Theorem 2. *Suppose a node receives data at rate $\mathbf{S_v}\lambda_{\mathbf{v}}$, and forwards at rate $\Gamma_v(\lambda_{\mathbf{v}})$. The profit $P_v = \kappa\Gamma_v(\lambda_{\mathbf{v}}) - \omega\mathbf{S_v}\lambda_{\mathbf{v}}$ is maximum at the optimal rate of arrival $\hat{\lambda}_{\mathbf{v}}$ when ω/κ is set equal to ξ.*

Proof. Let $\hat{\lambda}_{\mathbf{v}}$ be the vector of optimal rate of arrival as defined in Theorem 1, and let the corresponding profit be $\hat{P}_v$. Next we show that $\hat{P}_v \geq P_v$. From the definition of profit,

$\hat{P}_v - P_v = \kappa(\Gamma_v(\hat{\lambda}_\mathbf{v}) - \Gamma_v(\lambda_\mathbf{v})) - \omega\mathbf{S_v}(\hat{\lambda}_\mathbf{v} - \lambda_\mathbf{v})$. Using steps similar to the ones used to derive (5.8), we get

$$\hat{P}_v - P_v \geq -\kappa(\xi\mathbf{S_v}\mathbf{h_v}) - \omega\mathbf{S_v}(\hat{\lambda}_\mathbf{v} - \lambda_\mathbf{v})$$
$$\Rightarrow \hat{P}_v - P_v \geq (-\kappa\xi + \omega)\mathbf{S_v}\mathbf{h_v}$$

When $\omega/\kappa = \xi$, we get $\hat{P}_v - P_v \geq 0 \Rightarrow \hat{P}_v \geq P_v$. □

5.5.4 Collusion Resistance

Source, relays, and destination can collude with each other to increase their profit, e.g., a source can give destination a behind–the–scene compensation to not acknowledge the received packet (as a result, the source will not have to pay for the reward of relays). The possibilities of such collusions are analyzed in [91] which provides a number of conditions to ensure collusion resistance. In CRISP, the reward of relaying data when the packet is not delivered is zero, which corresponds to $\gamma = 0$ in [91]. Based on analysis in [91], the only additional requirement to make CRISP collusion-resistant to such manipulations is to ensure that the source makes the same fixed payment to the network irrespective of whether the packet is delivered or not. When the packet is delivered, each node gets a reward based on the utilities at which it received and forwarded the data, otherwise the payment stays in the network. However, the analysis in [91] does not consider collusion among relays to add fake relays or to change routing utility. CRISP secures against these collusions (see Section 5.2) as adding fake relays does not increase profit, and changing routing utility reduces profit.

5.6 Distributed Payment Framework (DPF)

Each relay pays ω to the previous relay in the path in order to receive a packet. If the packet is delivered, the source makes the final payment to the relays and the destination; otherwise, the source pays to the network. To enforce this payment scheme, any of the existing frameworks [60, 94] can be used. Here, we discuss a completely decentralized framework, without any requirement of an offline bank. Relaxing the requirement of an offline bank is useful for maintaining credits on sensor nodes and devices that do not connect to a central server often enough to validate earned credits.

Existing schemes have an online or an offline bank, which is a trusted authority. Each node submits all the transaction receipts or a summarized report of all transactions to the bank which can then check if a node is cheating and revoke its credits or its certificate. The problem becomes very challenging when such a trusted bank is not available to check all the receipts. A trivial solution is to keep track of all transaction receipts on each and every node, but it is prohibitive due to the large overhead of maintaining these receipts on every node.

5.6.1 Forged Credits

To maintain credit without a centralized bank, we propose a novel solution based on a simple idea – *credit is neither created nor destroyed*; it is only transferred between nodes. Each new node v is paid (initialized) a fixed amount of credit when it joins the network. A node then earns/pays credit depending on its activity in the network. Each node v also keeps track of initialized credit, paid credit and earned credit at every other node w in the network. In order to validate credit at any other node w, node v needs to confirm the initialized value of credit at node w, credit earned by node w, and credit paid by node w. A protocol to confirm all this information is presented in the next section. The key underlying intuition is that since credit

cannot be created, the credit earned by node w must have been paid by some other set of nodes x in the network. To validate all this information with low overhead in an opportunistic network is a major challenge – a solution is presented next.

5.6.2 Protocol

Each node that joins a network is initialized with some value of credit. A node's credit value changes as it generates data packets or receives/forwards data packets i.e. when a node makes payments to other nodes or earns rewards from other nodes. The payment is exchanged for each data packet. To make the protocol more efficient, nodes can also make a single payment for a net exchange of packets. Therefore, we describe our protocol in terms of how a given amount of payment is made between any two nodes. Next, we describe the different types of credits maintained at a node and how the credits records are exchanged to enable a distributed payment framework (DPF).

5.6.2.1 Types of credits

The credit record of a single node v is a 4-tuple $H_v := <I_v, \mathbf{Y_v}, \mathbf{R_v}, C_v>$. A node v is paid a credit value I_v when it joins the network. It makes payments represented by vector $\mathbf{Y_v} = (Y_{vw}, \forall w \in V)$ where Y_{vw} is the payment made by node v to node w and Y_{vv} is the payment node v must pay to the network as defined in CRISP. Similarly, node v receives payments from other nodes represented by vector $\mathbf{R_v} = (R_{vw}, \forall w \in V)$ where R_{vw} is the payment received by node v from node w and $R_{vv} := 0$. Finally, C_v is the current value of credit based on the value of initialized credit and payments made/received.

5.6.2.2 Transaction

A transaction is when a node makes or receives a payment. Here we only describe the method of making/receiving a payment. The need for making or receiving a payment can be triggered by any of the following events: a packet is generated, received, forwarded, or delivered. The rules for maintaining $\mathbf{Y_v}$, $\mathbf{R_v}$, and C_v are as follows:

1. Initially, when node v joins the network, $\mathbf{Y_v} := \mathbf{0}$ and $\mathbf{R_v} := \mathbf{0}$.
2. If node v needs to make a payment of amount ω to node w, it sets $Y_{vw} := Y_{vw} + \omega$
3. If node v needs to receive a payment of amount κ from node w, it sets $R_{vw} := R_{vw} + \kappa$
4. The nodes then updates its current value of credit C_v to be

$$C_v = I_v + \sum_{w \in V} R_{vw} - \sum_{w \in V} Y_{vw}.$$

Notice that the values Y_{vw} and Rvw always increase for all node pair $v, w \in V$. Thus, whenever node v finds out that node w has made additional payment i.e. $Y_{wv} > R_{vw}$, then node v receives additional payment (equal to $Y_{wv} - R_{vw}$) by setting $R_{vw} := Y_{wv}$.

5.6.2.3 Credits maintenance

Credit records of a node change after each transaction (i.e. when a node makes or receives a payment). To keep track of credits in a consistent manner, each node v also assigns a transaction number $L[v]$ to its credit record H_v. The transaction number is initialized at zero and is incremented by one after each transaction.

The credit record of each node is also flooded to the rest of the network. To reduce network overhead, each node can skip sharing some of its credit records (e.g. a node can share every 50th credit record) – as a result the exchange of credit between nodes takes longer but the payments still remain consistent as described in Section

5.6.3. Also, due to large delays in opportunistic network, information about most recent transaction is not available at every node at the same time. Thus, for the credit record of node v, every node w keeps the transaction number of the most recent record of node v that they have received. This is represented by the ordered pair $(H_v, L_w[v])$. For example, node v has completed 10 transactions – $L_v[v] = 10$, but node w has the credit record of the 7th transaction only – $L_w[v] = 7$. When any node v meets another node w and finds a greater transaction number for the credit record of some node x (i.e. $L_w[x] > L_v[x]$), node v updates the credit record of node x by receiving new data from node w.

Thus, each node v maintains credit records for every node w in the network. In particular, node v maintains ordered pairs $(H_w, L_v[w]), \quad \forall w \in V$ where $L_v[w]$ is the transaction number of node w for which node v has received the credit record H_w of node w.

5.6.2.4 Transaction numbers and vector clocks

The transaction numbers are effectively same as vector clocks [35, 61]. A vector clock is a vector of n integers where $n = |V|$ is the total number of nodes in the network V. Each node keeps its own vector clock L_v – in our case, $L_v[w]$ is the transaction number for credit record of node w that has been received at node v. More formally, the rules for updating transaction number (vector clocks) and credit records are as follows:

1. Initially, $L_v[w] = 0, \ \ \forall v, w \in V$
2. Just before node v makes a transaction [Section 5.6.2.2], it sets $L_v[v] := L_v[v]+1$.
3. Node v sends vector L_v when it connects with another node.
4. When node v receives vector L_w from node w:

(a) Node v receives H_x from the node w (to update its local record H_x) for all $x \in V$ with property $L_w[x] > L_v[x]$; and

(b) Node v sets

$$L_v[x] := max(L_v[x], L_w[x]), \quad \forall x \in V.$$

Thus, when node v comes into contact with any node w, node v updates the record H_x for all nodes x with property $L_w[x] > L_v[x]$.

5.6.3 Consistency

Credit records maintained at any node are typically not up to date due to large delays in the opportunistic network. However, a node still needs to validate if any other node is cheating and forging fake credit. The current credit of a node depends on three factors: initialized credit, payments made, and payments received. Initialized credit is set to be a fixed value for the entire network. Thus no individual node can change this value to claim more credit. A node can potentially make less payments than required to claim a higher value of credit but this is made impossible by use of Merkel trees as described in Section 5.6.4. Finally, a node can claim to have received payments that have not actually been made. To this end, we prove that following our protocol, it is guaranteed that records at any node will show proof of all payments that have been received irrespective of delays between connection among any pair of nodes.

Theorem 3. *Under DPF (distributed payment framework), after every transaction, local credit records at all nodes satisfy following property*

$$Y_{vw} \geq R_{wv} \quad \forall v, w \in V,$$

i.e. in the local set of records of any node, payments that are received are always less than or equal to the payments that are made.

Proof. A *happened–before* relationship means that a data packet cannot be received before it is sent (see vector clocks [35, 61]). For this reason, a node cannot receive payment before it is made. Next we show that the *happened–before* relationship is satisfied in the local set of records kept at all nodes.

Consider the record of credits maintained at node x. Consider node v pays node w at transaction number $\mathbf{A}[v]$ where $\mathbf{A} := \mathbf{L_v}$ is the transaction vector. This payment is received at node w at transaction number $\mathbf{B}[w]$ where $\mathbf{B} := \mathbf{L_w}$ is the transaction vector. Due to the *happened–before* relationship,

$$\mathbf{B}[v] \geq \mathbf{A}[v]. \tag{5.9}$$

Consider that records of credits maintained at node x show that this payment has been received at node w i.e.

$$L_x[w] \geq \mathbf{B}[w]. \tag{5.10}$$

Then, we need to prove that node x must also have the record that the payment has been made i.e. $L_x[v] \geq \mathbf{A}[v]$.

From (5.10), we have $L_x[w] \geq \mathbf{B}[w]$. From this result and the update rule (rule 4b) of transaction numbers, we have

$$\mathbf{L_x} \geq \mathbf{B} \text{ because } L_x[w] \geq \mathbf{B}[w].$$

This means that $L_x[v] \geq \mathbf{B}[v]$. From (5.9), $\mathbf{B}[v] \geq \mathbf{A}[v]$. Combining the previous two relationships, we get

$$L_x[v] \geq \mathbf{A}[v].$$

Thus, the local records at node x are definitely recent enough to validate that the the payment was made at node v.

Since, in the local records of any node, no payment can be shown to have been received without proof of payment at another node, local records satisfy the following property:

$$Y_{vw} \geq R_{wv} \quad \forall v, w \in V, \tag{5.11}$$

i.e. in the local set of records of any node, payments that are received are always less than or equal to the payments that are made. □

The payment mechanism is designed such that when credit record of any transaction is available, the credit records of all past transactions can be discarded. As an alternative to above proof, it can be shown that credits records of all nodes that are locally maintained at every node make the *frontier* of a *consistent cut* of the *global state* of transactions [25,30]. This is the key property that enables a node to validate credit at other nodes (by confirming received payments are less than payments made) even when information about most recent transaction is not available.

5.6.4 Security

To secure the credit records against tampering by other nodes, each node v signs its credit record $(H_v, L_v[v])$ with its private key after every transaction. This ensures that credit record of any node cannot be changed by any other node without knowledge of its private key.

To ensure a node's payment to the network when the packets are not delivered, a Merkel tree is used. In case of a delivered packet, a source *has* to make a payment, because relays can identify the source of the packet, and upon not receiving the payment they can discontinue providing the service for a non–paying source. We do not provide details of the Merkel trees here as a similar use for maintaining offline

credit is proposed in [60, 94]. The difference is the leaf value of the tree – in our case it is the hash of the packet that a source generates. The root value of tree is flooded in the network by each source. To ensure that a source makes the payment, relays request the *support* of the hash tree (see [60, 94]) for all the packets that the relay forwarded. The support of any data packet that a relay forwards requires checking $\log(m)$ constant–size (e.g. 160–bit) hashes, where m is the number of variable–sized packets that are leaves of the Merkel tree.

Each node maintains two types of Merkel trees. A *type 1* Merkel tree is used to keep track of all the data packets a node generates, and a *type 2* Merkel tree is used to keep track of all the *undelivered* data packets a node generates. To reduce overhead of checking, instead of verifying support of every data packet that a relay forwards, relays request the support of those data packets (from the type 2 Merkel tree) for which it did not receive the payment (i.e. potentially undelivered data packets). Upon verifying that all undelivered data packets are part of the type 2 Merkel tree, each node confirms that the corresponding payment has been made to the network. The payment to the network is stored in the credit record Y_{vv} for each node $v \in V$. The amount of payments is basically the number of leaf nodes of the type 2 Merkel tree multiplied by the fixed payment amount for each undelivered data packet. Once the total payment to the network by all nodes (equal to $\sum_{v \in V} Y_{vv}$) is greater than a certain threshold, this payment is uniformly redistributed to every node in the network. Also, to keep the Merkel tree from growing too large over time, a limit is set on the number of leaves (e.g. 512 leaves), and once it is reached a new Merkel tree is created. The root values of new and old Merkel trees are flooded in the entire network.

5.7 Evaluation

A relay pays ω credit to receive a packet and earns κ credit upon forwarding a packet under the CRISP scheme. It has been shown in several studies that cheating in the routing and forwarding stage results in poor network performance [28]. Therefore, we consider a special type of cheating that is possible in the existing credit schemes. Overall, we consider three types of behavior:

- Honest: The relay truthfully follows the routing and forwarding protocol.
- Selfish: The relay modifies the routing utility to operate at the optimal rate [Theorem 1].
- Cheating: The relay modifies the routing utility to maximize the forwarding rate.

Existing schemes promote cheating (a node maximizes its profit by maximizing the forwarding rate) and therefore suffer from reduced network throughput. In our evaluation, we show that,

1. Cheating reduces the network throughput compared to honest behavior.
2. CRISP discourages cheating and promotes honest and selfish behavior.
3. Selfish behavior result in similar throughput compared to honest behavior.

CRISP promotes honest behavior and beneficial selfish behavior (by taking into account the packets received in addition to the packets forwarded) and thereby maintains good throughput. We show all of the above properties hold for a wide range of ξ values, packet expiration times, and packet generation rates in both real and synthetic mobility environments.

5.7.1 Simulation setup

Extensive simulations are run on real and synthetic mobility traces. The real trace was collected from people walking in a state fair [74], and the synthetic trace

is generated from a Levy walk mobility model [73]. Data packets are generated at each node for a random destination. The real mobility trace has been collected from participants that carry GPS receivers which log position at 30 second intervals. In order to make a comparison with a suitable number of nodes, track logs from each day are considered to be a separate user. These logs have durations from around one to ten hours each day. We truncate all logs into a 90-minute span, during which 18 users record their location. Synthetic traces using the Levy walk model were generated with the settings recommended in [73] with an area of $500 \times 500m^2$. All plots show the average of five simulation runs and the confidence interval is plotted on the first standard deviation. We use temporal distance [83] as described in Chapter 3 as the routing scheme in simulation. All values of profit are normalized by the profit of a selfish node and the value of throughput is normalized by the value of an honest node.

5.7.2 Collusion Resistance

To make CRISP collusion-resistant to addition of fake relays, reward is based on the total improvement a node makes in the path towards the destination as described in Section 5.2.3. Figure 5.2 shows that the expected reward based on improvement in utility is linear in the number of packets that a node forwards in different sets of environments. A reward based on improvement is equivalent to a reward based on number of packets that a node forwards. Thus, our analysis on inventive compatibility is applicable when the reward is based on improvement. The value of utility is an approximation for improvement that needs to be made to deliver the packet. We study the routing metric of temporal distance as is used in [59, 83] and compute the utility by extending a relation derived in Chapter 3. Thus, for a temporal distance t, the utility is computed as e^{-kt}, where k depends on the diffusion of the node (we

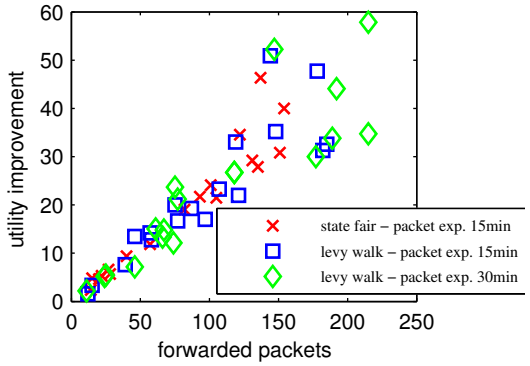

Figure 5.2 Improvement in utility is proportional to number of forwarded packets

take a value of 0.4/min, e.g., a 0.36 utility at a temporal distance of 2.5 mins). Thus, a node that receives a packet at $t = 10$ min and forwards it to a node at $t = 8$ min, receives an award of $e^{-0.32} - e^{-4}$ multiplied by the appropriate constant that makes the expected reward equal to κ.

5.7.3 Value of Threshold Constant ξ

In order to find the actual value of ξ that maximizes throughput, we use the queue approximation result described in Section 5.4.1. Let ψ be the average forwarding ratio among all nodes, i.e. the ratio of total packets forwarded to the total packets received over the whole network. From (5.7), the value of ξ is equal to $\frac{\partial \Gamma_v^x(\lambda_v^{x\star})}{\partial \lambda_v^x}$ where $\lambda_v^{x\star}$ is the solution for the equation $\frac{\Gamma_v^x(\lambda_v^{x\star})}{\lambda_v^{x\star}} = \psi$. We only need to know the value of δ_v^x to solve this equation[3]. In an actual network, the forwarding ratio of individual nodes is much different from the average value ψ. However, the arrival rates of all individual nodes still follow conditions of Theorem 1 for a constant value of ξ computed using the average value of ψ.

[3]The value of c_v^x is not required because ξ effectively depends on the ratio $\lambda_v^x/c_v^x = \rho_v^x$, i.e., $\xi = \frac{\partial(\rho_v^{x\star}(1-Q_v^x))}{\partial \rho_v^x}$ where $\rho_v^{x\star}$ is solution of equation $\frac{\rho_v^{x\star}(1-Q_v^x)}{\rho_v^{x\star}} = \psi$.

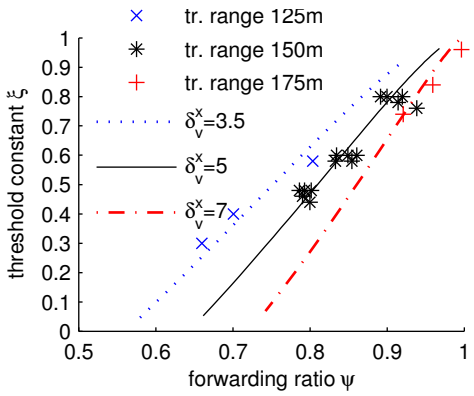

Figure 5.3 Value of threshold constant ξ – simulation and analysis

From simulation, we find ψ and the value of ξ that maximizes the throughput. Figure 5.3 compares the results from simulation and analysis. The lines show the mapping of forwarding ratio ψ to theoretically computed values of ξ at three different values of δ_v^x. The scatter points show the actual value ξ that maximizes throughput at the corresponding forwarding ratio ψ. Changing the generation rate or the packet expiration time changes the forwarding ratio ψ at nodes. In a specific environment, the value of ξ with different forwarding ratios (generation rate or the packet expiration time) lies close to the line corresponding to a fixed value of δ_v^x. Changing the environment from being sparse (transmission range 125m, forwarding ratio 65%) to more connected (transmission range 175m, forwarding ratio 98%) corresponds to changing the value of δ_v^x from 3.5 to 5. Thus, at a given forwarding ratio ψ, using the values of $\delta_v^x \in [3.5, 5]$ one can estimate a range such that a ξ that maximizes throughput belongs to that range. Later we show that CRISP is robust against inaccurate estimation of ξ.

5.7.4 Performance Across Nodes

In Fig. 5.4 and Fig. 5.5, each point corresponding to a node id is a complete simulation run in which that node's behavior is honest, selfish, or cheating. Figure 5.4

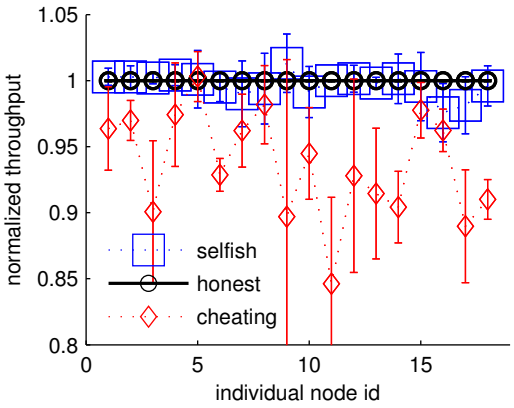

Figure 5.4 A single cheating node decreases network throughput by up to 15% (node #11).

shows that the network throughput when a node is honest or selfish is very close, but drops sharply when a node cheats. For example, the network throughput decreases by 15% when node #11 is cheating compared to the network throughput when node #11 is honest or selfish. The throughput degradation depends highly on the location and contacts of the node. For this reason, we show separate results for each of the cheating nodes. When two or more nodes cheat, then depending on their individual impact, they can drop a large percentage of data. For example, if node #1 and #2 cheat, they may only decrease the throughput by 5%, whereas if node #9 and #11 cheat, they can drop up to 25% of total data as can be seen in Fig. 5.4.

Figure 5.5 shows that the profit under CRISP of a selfish node is highest, a honest node is very close to it, and a cheating node is much lower (mostly negative). For example, when node #11 cheats, its profit is 500% less than when it is honest or selfish. The exception is when node #5 or #15 cheat, they make more profit than honest, but still less than selfish. This is not a problem because the throughput of selfish behavior is very close (same for node #5, slightly lower for node #15) to that of honest behavior as shown in Fig. 5.4.

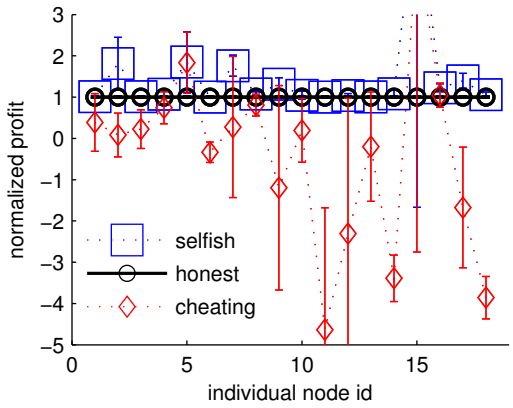

Figure 5.5 CRISP reduces the profit of cheating by up to 500% (node #11)

5.7.5 Robustness and Incentive Compatibility

The value of ξ that maximizes throughput is 0.65 in Fig. 5.6 and 5.7 for the state fair trace. These figures show the results averaged over all nodes such that in one simulation run only a single node changes its behavior (honest, selfish, or cheating). Figure 5.7 shows that profit under CRISP of honest (and selfish) behavior is greater than cheating for all values of threshold constant $\xi > 0.4$. Also, for all values of ξ, Fig. 5.6 shows that throughput of selfish (and honest) behavior is greater than cheating. In real time, a node cannot figure out the exact utility it should declare for selfish or cheating behavior. In our simulations, we try out different (constant) shifts in the routing utility and plot the results only for the change in routing utility that maximizes forwarding rate (cheating) or profit under CRISP (selfish). Thus, CRISP is quite robust to inaccurate values of ξ and maintains incentive compatibility for a range (0.5 to 0.7) of ξ values because a node is guaranteed a good profit just by honestly routing and forwarding.

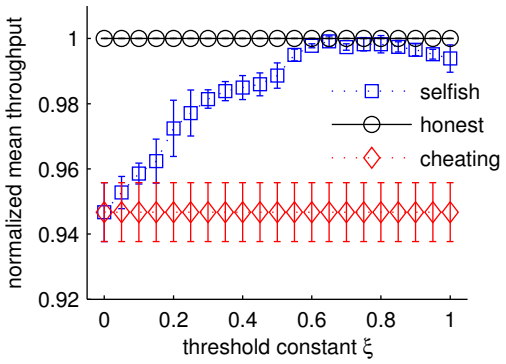

Figure 5.6 CRISP is robust over different values of threshold constant ξ – at all values of ξ throughput with selfish behavior is greater than that of cheating

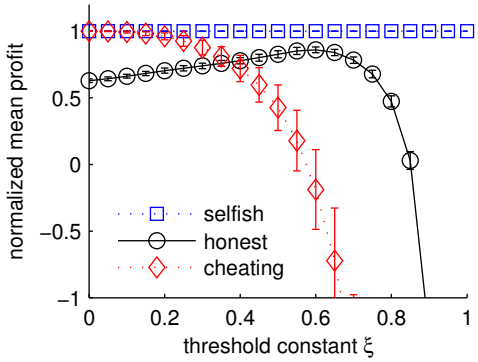

Figure 5.7 CRISP is incentive compatible over different values of threshold constant ξ – at values $\xi > 0.4$ profit of selfish (or honest) behavior is greater than cheating

5.7.6 Performance Across Packet Expiration Times and Packet Generation Rates

Figures 5.8 and 5.9 show mean throughput when packet expiration time and packet generation rates are varied in the state fair mobility trace. Each point corresponds to results averaged over 90 simulation runs (5 for each of 18 individual nodes when that node is honest, selfish, or cheating). It is evident from Fig. 5.8 and 5.9 that the network throughput with selfish nodes is within 1% of that of the honest nodes whereas that of cheating nodes is around 5% lower. Note that the throughput is averaged over all nodes such that in one simulation only a single node changes its behavior (honest, selfish, or cheating). Thus, the impact of multiple cheating nodes is

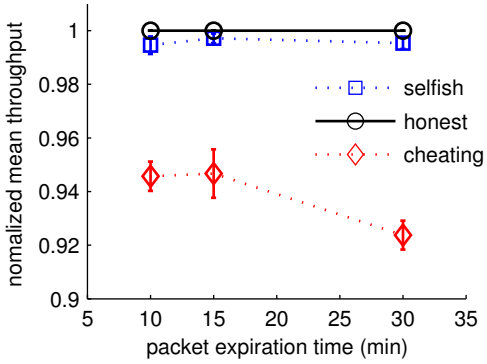

Figure 5.8 Throughput across different packet expiration times

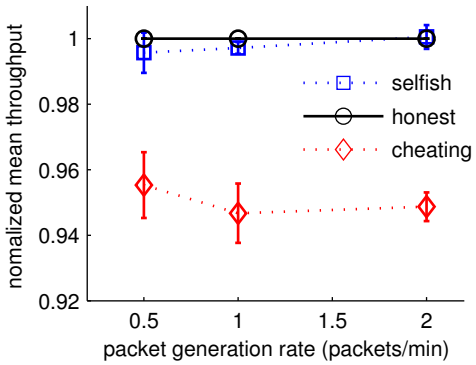

Figure 5.9 Throughput across different packet generation rates

much higher than 5%. Also, at a higher packet expiration time of 30 min, a cheating node attracts more packets than it does at the expiration time of 15 min and less. Therefore, the throughput of the network decreases from 5% to 8% with high packet expiration times.

5.8 Summary

In this chapter, we proposed CRISP, a collusion–resistant credit scheme designed to ensure profit maximization when a node behaves honestly when receiving and forwarding packets. Existing credit schemes reward upon forwarding packets which reduces the network throughput when a node tries to maximize its forwarding

rate. In comparison, CRISP optimizes a specific packet arrival and forwarding rate by adjusting the ratio of reward to payment (for forwarding and receiving packets respectively). CRISP secures against collusion of nodes to change routing utility or add fake relays in the path and against both flooding and blackhole attacks. Extensive simulation results demonstrate that CRISP promotes honest (or selfish) behavior which maintains good throughput. CRISP is shown to be robust with varying mobility characteristics, packet generation rates, packet expiration times and threshold constant. Furthermore, we provide the first consistent and secure payment framework that enables implementation of CRISP in an opportunistic network and does not require a trusted third party to maintain credits.

CHAPTER 6

CONCLUSION AND DIRECTIONS FOR FUTURE WORK

In this chapter we conclude the dissertation and provide directions for future research. This dissertation addresses key challenges to enable effective collaboration in opportunistic networks. Opportunistic networks are formed by connections between devices (like smartphones, sensors, etc.) carried by people. However, as people move around, new connections are formed whereas some connections are lost.

To collaborate effectively, a user needs to be able to send/receive data from devices in the environment even in the absence of an end–to–end connected path. Since there may not be a direct connection to exchange information in real–time, a user also needs a mechanism to select suitable devices providing the required services. If no single (nearby) device is providing the required service, there may exist a combination of services components on several devices that meets the requirements of the requested service (e.g. to meet the requirement of compression and encryption, one device compresses the file and another device encrypts the compressed file). This concept is formally known as service composition (i.e. combining/aggregating services). Performing service composition (which essentially requires transferring data between devices) is only possible through cooperation of other devices in the environment which provide resources (bandwidth, memory, computation time etc.) to *relay* data. Thus, to summarize, effective collaboration requires an efficient mechanism to transfer data, compose services and promote user participation. This dissertation addresses key challenges in the existing works to enable effective collaboration. These

challenges along with our proposed solutions and directions for future research are described next.

Existing routing metrics for data transfer exhibit efficient performance in either homogeneous (users have similar mobility characteristics) or specific heterogeneous (varied mobility characteristics) scenarios. However, in practice, behavior of users changes at different locations and times, making it hard to generalize any one routing algorithm. Therefore, an efficient and adaptive forwarding scheme, ProxiMol, is presented that improves message delivery ratio and reduces average delay in a range of environments with varied mobility characteristics and different levels of connectivity. To the best of our knowledge, this is the first work that derives a forwarding metric based on power-law distributions of inter-contact times, flight lengths and pause durations. These characteristics have been shown to be statistically significant in real mobility traces [73]. Based on our analysis and simulation studies, we conclude that *mobility* (measured by diffusion of a user) plays a central role in a highly heterogeneous network environment while a user's *proximity to destination* is useful in relatively homogenous settings in improving delivery ratio. Using statistical properties of real mobility traces, a model is proposed to approximate a user's location over time using a locally measured value of her diffusion. This model is used to define the key tradeoff between proximity and mobility to compare the likelihood of delivering data. An important and interesting direction for future research is to also take into account the direction of a user's movement in order to compare the likelihood of delivery. A number of methods can be useful in estimating user direction. User direction can be estimated relative to other nodes or it can be absolute. Some possibilities to make such estimates are by using GPS, landmarks (e.g. using signal strength of Wi–Fi access–points whose location is known), or information related to connection/disconnection with other devices in the environment.

Our second contribution is a novel algorithm for service composition in opportunistic networks. In existing works, service composition is not performed on remote devices in the absence of a connected path. However, this is directly resolved in our solution. As a result, mobile users can benefit from a larger set of services available locally in an environment. Proposed service composition makes efficient service selections from devices that are in close proximity (but not necessarily connected to the service requestor). Key insight used in the solution is that temporal distance between devices provides a direct measure for reachability of devices when an end-to-end connected path does not exist. This way, based on a service graph, a composition sequence can be selected to meet the requirements of service request while any routing scheme can be used to forward service parameters. Through extensive simulations on real and synthetic mobility traces, we conclude that using only local knowledge about temporal distances and service loads, a large percentage of services can be composed in an opportunistic network using multi-hop paths among devices. In scenarios when some services are requested more often than others, performance improves when less popular (requested) services are replaced by more popular (requested) services. Future works can explore this key insight about replacement of less popular services with more popular services - specifically based on location and mobility characteristics of the user providing a particular service. As a result, using the same (or possibly even a smaller) amount of resources, more service requests can be completed in a smaller duration of time.

Our final contribution is a collusion–resistant credit scheme, CRISP, designed to ensure profit maximization when a node behaves honestly to receive and forward data packets. Existing credit schemes reward upon forwarding packets which reduces the network throughput when a node tries to maximize its forwarding rate. In comparison, CRISP optimizes a specific packet arrival and forwarding rate by adjusting the

ratio of reward (for forwarding packets) to payment (for receiving packets). Extensive simulation results demonstrate that CRISP promotes honest (or selfish) behavior to maintain good throughput. CRISP is shown to be robust with varying mobility characteristics, packet generation rates, packet expiration times and threshold constant. Furthermore, we provide the first consistent and secure payment framework that enables implementation of CRISP in an opportunistic network and does not require a trusted third party to maintain credits. There are a number of critical directions in which this work can be extended. Future works can consider the effect of different policies/strategies for spending and earning credit and its effect on network performance - e.g. how to define allocation of available credit in two parts where one part is used as payment to generate data packets whereas the other part is used as payment to relay data of other nodes (in hopes of earning more credit). Future works can also consider payment requirements for applications in more involved scenarios that include multicast, broadcast, or replication.

APPENDIX A

PUBLICATIONS FROM THIS DISSERTATION

1. Umair Sadiq. Collaboration in Opportunistic Networks. In *Proc. 8th IEEE PERCOMW*, March 2011.
2. Umair Sadiq and Mohan Kumar. ProxiMol: Proximity and mobility estimation for efficient forwarding in opportunistic networks. In *Proc. 8th IEEE MASS*, Oct 2011. Second best student paper award – out of 244 submissions.
3. Umair Sadiq, Mohan Kumar, Andrea Passarella and Marco Conti. Modeling and Simulation of Service Composition in Opportunistic Networks. In *Proc. 14th ACM MSWiM*, Oct 2011. Nominated for best paper award – among top 3 of 190 submissions.
4. Umair Sadiq, Mohan Kumar, and Matthew Wright. CRISP: Collusion–Resistant Incentive–Compatible Routing and Forwarding in Opportunistic Networks. In *Proc. 15th ACM MSWiM*, Oct 2012.
5. Umair Sadiq, Mohan Kumar, Andrea Passarella and Marco Conti. Service Composition in Opportunistic Networks: A Load and Mobility Aware Solution. Journal, in progress.
6. Umair Sadiq, Mohan Kumar, and Matthew Wright. CRISP: Collusion–Resistant Incentive–Compatible Routing and Forwarding in Opportunistic Networks. Journal, in progress.

REFERENCES

[1] A community resource for archiving wireless data at dartmouth. Cited on page 27.

[2] Consumer generated mobile wireless media. Cited on page 3.

[3] Delay tolerant networking research group. Cited on page 3.

[4] Situated and autonomic communications. Cited on page 3.

[5] `http://fif.kr/fisc2009/doc/khlee.pdf`, 2009. Cited on page 73.

[6] Luzi Anderegg and Stephan Eidenbenz. Ad hoc-vcg: A truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *ACM MobiCom*, 2003. Cited on pages 17, 18, and 20.

[7] Mustafa Y. Arslan, Konstantinos Pelechrinis, Ioannis Broustis, Srikanth V. Krishnamurthy, Prashant Krishnamurthy, and Prasant Mohapatra. Detecting route attraction attacks in wireless networks. In *Proc. IEEE MASS*, 2011. Cited on page 20.

[8] Aruna Balasubramanian, Brian Levine, and Arun Venkataramani. Dtn routing as a resource allocation problem. *SIGCOMM Comput. Commun. Rev.*, 37(4):373–384, 2007. Cited on pages 22 and 41.

[9] Aruna Balasubramanian, Ratul Mahajan, and Arun Venkataramani. Augmenting mobile 3g using wifi. In *Proc. ACM MobiSys*, 2010. Cited on page 2.

[10] D. Y. Barrer. Queuing with impatient customers and ordered service. *Operations Research*, 5(5):pp. 650–656, 1957. Cited on pages 93 and 94.

[11] C. Boldrini and A. Passarella. Hcmm: Modelling spatial and temporal properties of human mobility driven by users social relationships. *Computer Communications*, 33(9):1056–1074, 2010. Cited on pages 51 and 52.

[12] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University, 2004. Cited on pages 94 and 96.

[13] J. Burgess, G. D. Bissias, M. D. Corner, and B. N. Levine. Surviving attacks on disruption-tolerant networks without authentication. In *Proc. ACM MobiHoc*, 2007. Cited on page 19.

[14] John Burgess, Brian Gallagher, David Jensen, and Brian Neil Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proc. IEEE INFOCOM*, 2006. Cited on pages 5, 6, 34, 41, 85, and 86.

[15] B. Burns, O. Brock, and B.N. Levine. Mv routing and capacity building in disruption tolerant networks. In *Proc. IEEE INFOCOM*, 2005. Cited on page 14.

[16] L. Buttyan and J.P. Hubaux. Enforcing service availability in mobile ad-hoc wans. In *ACM MobiHoc*, 2000. No cited.

[17] L. Buttyan and J.P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM MoNet*, 8(5):579–592, 2003. Cited on pages 17 and 18.

[18] M. Conti C. Boldrini and A. Passarella. Contentplace: Social-aware data dissemination in opportunistic networks. In *Proc. MSWiM*, 2008. Cited on page 1.

[19] Han Cai and Do Young Eun. Crossing over the bounded domain: from exponential to power-law inter-meeting time in manet. In *Proc. ACM MobiCom*, 2007. Cited on page 22.

[20] Han Cai and Do Young Eun. Aging rules: what does the past tell about the future in mobile ad-hoc networks? In *Proc. ACM MobiHoc*, 2009. Cited on pages 13, 45, and 57.

[21] S. Capkun, L. Buttyan, and J. Hubaux. Sector: Secure tracking of node encounters in multi-hop wireless networks. In *Proc. ACM SASN*, 2003. Cited on pages 8, 19, and 20.

[22] Iacopo Carreras, Louay Bassbouss, David Linner, Heiko Pfeffer, Vilmos Simon, Endre Varga, Daniel Schreckling, Jyrki Huusko, and Helena Rivas. Bionets: Self evolving services in opportunistic networking environments. In *Bioinspired Models of Network, Information, and Computing Systems*, pages 88–94. 2010. Cited on pages 7, 15, and 47.

[23] Augustin Chaintreau, Pan Hui, Jon Crowcroft, Christophe Diot, Richard Gass, and James Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6:606–620, 2007. Cited on pages 5, 23, and 50.

[24] Dipanjan Chakraborty, Anupam Joshi, Tim Finin, and Yelena Yesha. Service composition for mobile environments. *Mobile Networks and Applications*, 10:435–451, 2005. Cited on page 16.

[25] K.M. Chandy and L. Lamport. Distributed snapshots: determining global states of distributed systems. *ACM Transactions on Computer Systems (TOCS)*, 3(1):63–75, 1985. Cited on page 108.

[26] Bin Bin Chen and Mun Choon Chan. Mobicent: a credit-based incentive system for disruption tolerant network. In *Proc. IEEE INFOCOM*, 2010. Cited on page 87.

[27] Tingting Chen and Sheng Zhong. Inpac: An enforceable incentive scheme for wireless networks using network coding. In *Proc. IEEE INFOCOM*, 2010. Cited on pages 8 and 87.

[28] Fai Cheong Choo, Mun Choon Chan, and Ee-Chien Chang. Robustness of dtn against routing attacks. In *Proc. COMSNETS*, 2010. Cited on pages 19 and 110.

[29] M. Conti and M. Kumar. Opportunities in opportunistic computing. *Computer*, 43(1):42–50, 2010. Cited on pages 1, 6, and 7.

[30] G.F. Coulouris, J. Dollimore, and T. Kindberg. *Distributed systems: concepts and design*. Addison-Wesley Longman, 2005. Cited on page 108.

[31] Elizabeth M. Daly and Mads Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proc. ACM MobiHoc*, 2007. Cited on pages 5 and 14.

[32] Oleg Davidyuk, Ivan Sanchez, and Jukka Riekki. CADEAU: Supporting Autonomic and User-Controlled Application Composition in Ubiquitous Environments. In *Pervasive Computing and Communications Design and Deployment: Technologies, Trends, and Applications*. 2010. Cited on pages 15 and 47.

[33] L. Del Prete and L. Capra. Reliable discovery and selection of composite services in mobile environments. In *Proc. IEEE EDOC*, 2008. Cited on pages 15, 47, and 66.

[34] Henri Dubois-Ferriere, Matthias Grossglauser, and Martin Vetterli. Age matters: efficient route discovery in mobile ad hoc networks using encounter ages. In *Proc. ACM MobiHoc*, 2003. Cited on pages 6, 13, 17, 32, 34, 37, and 57.

[35] C. Fidge. Logical time in distributed computing systems. *Computer*, 24(8):28–33, 1991. Cited on pages 105 and 107.

[36] Joy Ghosh, Sumesh J. Philip, and Chunming Qiao. Sociological orbit aware location approximation and routing (solar) in manet. *Ad Hoc Netw.*, 5(2):189–209, 2007. Cited on page 5.

[37] Martin Groβ and Martin Skutella. Generalized maximum flows over time. In *Matheon Preprint 878*, 2012. Cited on page 89.

[38] R. Groenevelt, E. Altman, and P. Nain. Relaying in mobile ad hoc networks: the brownian motion mobility model. *Wirel. Netw.*, 12(5):561–571, 2006. Cited on page 30.

[39] M. Grossglauser and M. Vetterli. Locating nodes with ease: last encounter routing in ad hoc networks through mobility diffusion. In *Proc. IEEE INFOCOM*, 2003. Cited on pages 6, 13, 17, 32, 34, 37, and 57.

[40] Xiaohui Gu, K. Nahrstedt, and Bin Yu. Spidernet: an integrated peer-to-peer service composition framework. In *Proc. IEEE HPDC*, pages 110–119, 2004. Cited on page 16.

[41] Bo Han, Pan Hui, V.S.A. Kumar, M.V. Marathe, Jianhua Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *IEEE Transactions on Mobile Computing*, 11(5):821–834, 2012. Cited on page 2.

[42] S. Heimlicher, M. Karaliopoulos, H. Levy, and T. Spyropoulos. On leveraging partial paths in partially-connected networks. In *Proc. IEEE INFOCOM*, 2009. Cited on pages 5, 6, 14, 21, and 44.

[43] Andreas Heinemann. *Collaboration in Opportunistic Networks*. VDM Verlag, 2007. Cited on page 1.

[44] B.D. Hughes. *Random Walks and Random Environments*. Oxford University Press, 1995. Cited on page 25.

[45] Pan Hui, Augustin Chaintreau, James Scott, Richard Gass, Jon Crowcroft, and Christophe Diot. Pocket switched networks and human mobility in conference environments. In *Proc. ACM SIGCOMM WDTN*, 2005. Cited on page 5.

[46] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *Proc. ACM MobiHoc*, 2008. Cited on pages 5, 7, 14, 85, and 86.

[47] 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, 2007. Cited on page 25.

[48] Swaroop Kalasapur, Mohan Kumar, and Behrooz A. Shirazi. Dynamic service composition in pervasive computing. *IEEE Transactions on Parallel and Distributed Systems*, 18:907–918, 2007. Cited on pages 6, 15, 16, 47, and 50.

[49] Sungwon Kim and Do Do Eun. Impact of super-diffusive behavior on routing performance in delay tolerant networks. In *Proc. IEEE ICC*, 2008. Cited on page 32.

[50] Sungwon Kim, Chul-Ho Lee, and Do Young Eun. Superdiffusive behavior of mobile nodes and its impact on routing protocol performance. *IEEE Transactions on Mobile Computing*, 9(2):288–304, 2010. Cited on pages 25, 26, 27, and 56.

[51] D. Lapsley and S. Low. An optimization approach to abr control. In *IEEE ICC*, 1998. Cited on page 95.

[52] Nicolas Le Sommer, Romeo Said, and Yves Mahéo. A proxy-based model for service provision in opportunistic networks. In *Proc. MPAC*. ACM, 2008. Cited on pages 15 and 47.

[53] Kyunghan Lee, Seongik Hong, Seong Joon Kim, Injong Rhee, and Song Chong. Slaw: A new mobility model for human walks. In *Proc. IEEE INFOCOM*, 2009. Cited on pages 23, 51, and 73.

[54] J. Leguay, T. Friedman, and V. Conan. Evaluating mobility pattern space routing for dtns. In *Proc. IEEE INFOCOM*, 2006. Cited on page 22.

[55] F. Li, A. Srinivasan, and J. Wu. Thwarting blackhole attacks in disruption-tolerant networks using encounter tickets. In *Proc. IEEE INFOCOM*, 2009. Cited on pages 8, 19, and 20.

[56] Na Li and Sajal K. Das. Radon: reputation-assisted data forwarding in opportunistic networks. In *Proc. MobiOpp*, 2010. Cited on pages 8, 19, and 20.

[57] Ben Liang and Zygmunt J. Haas. Predictive distance-based mobility management for multidimensional pcs networks. *IEEE/ACM Transactions on Networking*, 11(5):718–732, 2003. Cited on page 5.

[58] S. Lim, C. Yu, and C. R. Das. Clustered mobility model for scale-free wireless networks. In *Proc. IEEE LCN*, 2006. Cited on page 5.

[59] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. In *Proc. Workshop on Service Assurance with Partial and Intermittent Resources (SAPIR)*, 2004. Cited on pages 5, 6, 7, 13, 34, 37, 85, 86, and 111.

[60] Rongxing Lu, Xiaodong Lin, Haojin Zhu, Xuemin Shen, and B. Preiss. Pi: A practical incentive protocol for delay tolerant networks. *IEEE Transactions on Wireless Communications*, 9(4):1483–1493, 2010. Cited on pages 8, 19, 20, 88, 102, and 109.

[61] F. Mattern. Virtual time and global states of distributed systems. *Parallel and Distributed Algorithms*, 1(23):215–226, 1989. Cited on pages 105 and 107.

[62] Marvin McNett and Geoffrey M. Voelker. Access and mobility of wireless pda users. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(2):40–55, 2005. Cited on page 27.

[63] M. Musolesi and C. Mascolo. Car: Context-aware adaptive routing for delay-tolerant mobile networks. *IEEE Transactions on Mobile Computing*, 8(2):246–260, 2009. Cited on page 22.

[64] S.C. Nelson, M. Bakht, and R. Kravets. Encounter-based routing in dtns. In *Proc. IEEE INFOCOM*, 2009. Cited on pages 5, 6, 14, 17, 32, 33, 37, 41, and 52.

[65] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *Games and Economic Behavior*, pages 129–140, 1999. Cited on page 87.

[66] M. Onen, A. Shikfa, and R. Molva. Optimistic fair exchange in secure forwarding. In *Proc. MobiQuitous*, 2007. Cited on pages 8, 19, and 20.

[67] J. Ott, E. Hyytia, P. Lassila, T. Vaegs, and J. Kangasharju. Floating content: Information sharing in urban areas. In *Proc. IEEE PerCom*, 2011. Cited on pages 1 and 3.

[68] Andrea Passarella, Marco Conti, Eleonora Borgia, and Mohan Kumar. Performance evaluation of service execution in opportunistic computing. In *Proc. 13th ACM MSWIM*, 2010. Cited on pages 6, 16, and 47.

[69] Andrea Passarella, Mohan Kumar, Marco Conti, and Elenora Borgia. Minimum-delay service provisioning in opportunistic networks. *IEEE Transactions on Parallel and Distributed Systems*, 99(PrePrints), 2010. Cited on pages 6, 15, 16, 47, 51, and 59.

[70] L. Pelusi, A. Passarella, and M. Conti. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*, 44(11):134–141, 2006. Cited on page 6.

[71] A-K Pietilinen. *Opportunistic Mobile Social Networks at Work*. Phd, Universite Pierre et Marie Curie, Paris 6, Laboratoire d'Informatique, December 2010. Cited on page 3.

[72] Yanzhi Ren, Mooi Choo Chuah, Jie Yang, and Yingying Chen. Detecting black-hole attacks in disruption-tolerant networks through packet exchange recording. In *Proc. IEEE WoWMoM*, 2010. Cited on pages 8, 19, and 20.

[73] Injong Rhee, Minsu Shin, Seongik Hong, Kyunghan Lee, and Song Chong. On the levy-walk nature of human mobility. In *Proc. IEEE INFOCOM*, 2008. Cited on pages 5, 22, 23, 27, 33, 35, 36, 40, 45, 51, 63, 111, and 120.

[74] Injong Rhee, Minsu Shin, Seongik Hong, Kyunghan Lee, Seongjoon Kim, and Song Chong. CRAWDAD trace. `http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels/GPS/NC_State_Fair`, July 2009. Cited on pages 52, 58, 63, and 110.

[75] Natasa Sarafijanovic-djukic, Michal Piórkowski, and Matthias Grossglauser. hopping: Efficient mobility-assisted forwarding in partitioned networks. In *Proc. IEEE SECON*, 2006. Cited on pages 5, 6, and 21.

[76] Cigdem Sengul, Aline Carneiro Viana, Roy Friedman, Marin Bertier, and Anne-Marie Kermarrec. Adaptive Forwarding to Match Mobility Characteristics in Delay Tolerant Networks. Research report, INRIA, 2009. Cited on page 22.

[77] G. Sharma and R.R. Mazumdar. Scaling laws for capacity and delay in wireless ad hoc networks with random mobility. In *Proc. IEEE ICC*, 2004. Cited on pages 23 and 51.

[78] Maiko Shigeno. Maximum network flows with concave gains. *Mathematical Programming*, 107:439–459, 2006. Cited on page 89.

[79] Nicolas Le Sommer and Salma Ben Sassi. Location-based service discovery and delivery in opportunistic networks. In *Proc. ICN*, 2010. Cited on pages 15 and 47.

[80] T. Spyropoulos, T. Turletti, and K. Obraczka. Routing in delay-tolerant networks comprising heterogeneous node populations. *IEEE Transactions on Mo-*

bile Computing, 8(8):1132–1147, 2009. Cited on pages 5, 6, 14, 17, 32, 35, 37, and 52.

[81] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. ACM SIGCOMM WDTN*, 2005. Cited on pages 13, 14, and 42.

[82] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *Proc. IEEE PERCOMW*, 2007. Cited on pages 6, 13, 32, 34, 37, and 41.

[83] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Efficient routing in intermittently connected mobile networks: the single-copy case. *IEEE/ACM Transactions on Networking*, 16(1):63–76, 2008. Cited on pages 5, 6, 7, 13, 17, 32, 34, 37, 52, 56, 57, 60, 85, 86, and 111.

[84] John Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Characterising temporal distance and reachability in mobile and online social networks. *SIGCOMM Comput. Commun. Rev.*, 40(1):118–124, 2010. Cited on pages 32 and 57.

[85] László A. Végh. Concave generalized flows with applications to market equilibria. *CoRR*, 2011. Cited on page 89.

[86] J. Wang. Exploiting mobility prediction for dependable service composition in wireless mobile ad hoc networks. *IEEE Trans. on Services Comput.*, 4:44–55, 2011. Cited on pages 15, 47, and 66.

[87] Weizhao Wang, Stephan Eidenbenz, Yu Wang, and Xiang-Yang Li. Ours: optimal unicast routing systems in non-cooperative wireless networks. In *Proc. ACM MobiCom*, 2006. Cited on pages 8, 17, 18, and 20.

[88] Weizhao Wang and Xiang-Yang Li. Low-cost routing in selfish and rational wireless ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(5):596–607, 2006. Cited on pages 8, 20, and 87.

[89] Fan Wu, Tingting Chen, Sheng Zhong, Li Erran Li, and Yang Richard Yang. Incentive-compatible opportunistic routing for wireless networks. In *Proc. ACM MobiCom*, pages 303–314, 2008. Cited on pages 8 and 87.

[90] Quan Yuan, Ionut Cardei, and Jie Wu. Predict and relay: an efficient routing in disruption-tolerant networks. In *Proc. ACM MobiHoc*, 2009. Cited on page 14.

[91] S. Zhong, J. Chen, and Y.R. Yang. Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *Proc. IEEE INFOCOM*, 2003. Cited on pages 17, 18, 85, and 101.

[92] Sheng Zhong, Li Erran Li, Yanbin Grace Liu, and Yang Richard Yang. On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks: an integrated approach using game theoretic and cryptographic techniques. *Wirel. Netw.*, 13:799–816, December 2007. Cited on pages 8, 17, 18, 20, 86, and 87.

[93] Sheng Zhong and Fan Wu. On designing collusion-resistant routing schemes for non-cooperative wireless ad hoc networks. In *Proc. ACM MobiCom*, 2007. Cited on pages 8 and 20.

[94] Haojin Zhu, Xiaodong Lin, Rongxing Lu, Yanfei Fan, and Xuemin Shen. Smart: A secure multilayer credit-based incentive scheme for delay-tolerant networks. *IEEE Transactions on Vehicular Technology*, 58(8), 2009. Cited on pages 8, 19, 20, 84, 88, 102, and 109.

BIOGRAPHICAL STATEMENT

Umair Sadiq received his degree of Bachelor of Science in Electronic Engineering from Ghulam Ishaq Khan Institute in May 2007, graduating at the top of his class. He was the recipient of Faculty Medal for Best Academic Performance in Electronic Engineering, Ghulam Ishaq Khan Medal for Overall Best Academic Performance in All Faculties, and Gold Medal for Best All-round Student in Academics, Extracurricular and Sports among engineering colleges in Pakistan – awarded by President of Pakistan Gen. Pervez Musharraf at Presidential House in 2007. He also won the National Physics Contest in Pakistan and received a Bronze Medal at 34th International Physics Olympiad. He worked as an Instrumentation Engineer at AkzoNobel for a year and then entered graduate school at The University of Texas at Arlington in 2008. He started working under the supervision of Professor Mohan Kumar and became a member of the Pervasive and Invisible Computing Group. He was admitted to PhD candidacy in 2011. The same year he also received the second best student paper award at 8th IEEE MASS conference and a nomination for best paper award at 14th ACM MSWiM conference. He received the Outstanding Research Award in the Department of Computer Science and Engineering at The University of Texas at Arlington in 2012.