Effective and Efficient Communication and Collaboration in Participatory Environments

Shiraz Qayyum

EFFECTIVE AND EFFICIENT COMMUNICATION AND COLLABORATION IN PARTICIPATORY ENVIRONMENTS

by

SHIRAZ QAYYUM

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in Computing and Information Sciences

B. Thomas Golisano College of Computing and
Information Sciences

Rochester Institute of Technology
Rochester, New York
May 6, 2015

EFFECTIVE AND EFFICIENT COMMUNICATION AND COLLABORATION IN PARTICIPATORY ENVIRONMENTS

by

SHIRAZ QAYYUM

Committee Approval:

We, the undersigned committee members, certify that we have advised and/or supervised the candidate on the work described in this dissertation. We further certify that we have reviewed the dissertation manuscript and approve it in partial fulfillment of the requirements of the degree of Doctor of Philosophy in Computing and Information Sciences.

Dr. Mohan Kumar Ph.D. Supervising Professor	Date
Dr. Minseok Kwon Ph.D. Committee Member	Date
Dr. Sumita Mishra Ph.D. Committee Member	Date
Dr. Tae Oh Ph.D. Committee Member	Date
Dr. Mihail Barbosu Dissertation Defense Chair	Date

Certified by:

Dr. Pengcheng Shi Director, Computing and Information Sciences	Date

ACKNOWLEDGEMENTS

First and foremost I would like to thank my supervising professor Dr. Mohan Kumar for constantly motivating and encouraging me. I appreciate all his contributions of time, ideas, and funding to make my Ph.D. experience productive and stimulating.

I wish to thank my committee members Dr. Minseok Kwon, Dr. Sumita Mishra and Dr. Tae Oh for their interest in my research and for taking time to serve in my dissertation committee.

I would also like to thank Dr. Matthew Wright, Dr. Donggang Liu, Dr. Yonghe Liu and David Levine for being on my dissertation committee when I was affiliated with The University of Texas at Arlington. I am especially grateful to Dr. Umair Sadiq for his interest in my research and for the helpful discussions and invaluable comments. I also wish to thank Dr. Sajal Das, Dr. Peizhao Hu, Dr. Daisuke, Mehrab and Swathy for taking the time to critically evaluate this manuscript.

Lastly, I would like to thank my parents and sisters for all their love, encouragement and support. And most of all, the faithful support from my loving, encouraging and patient wife Qurrat Ul Ain during the final stages of this Ph.D. is appreciated. Thank you.

6, May , 2015

ABSTRACT

EFFECTIVE AND EFFICIENT COMMUNICATION AND COLLABORATION IN PARTICIPATORY ENVIRONMENTS

SHIRAZ QAYYUM, Ph.D.

Rochester Institute of Technology, 2015

Supervising Professor: Dr. Mohan Kumar

Participatory environments pose significant challenges to deploying real applications. This dissertation investigates exploitation of opportunistic contacts to enable effective and efficient data transfers in challenged participatory environments.

There are three main contributions in this dissertation:

1. A novel scheme for predicting contact volume during an opportunistic contacts (PCV);
2. A method for computing paths with combined optimal stability and capacity (COSC) in opportunistic networks; and
3. An algorithm for mobility and orientation estimation in mobile environments (MOEME).

The proposed novel scheme called PCV predicts contact volume in soft real-time. The scheme employs initial position and velocity vectors of nodes along with the data rate profile of the environment. PCV enables efficient and reliable data transfers between opportunistically meeting nodes.

The scheme that exploits capacity and path stability of opportunistic networks is based on PCV for estimating individual link costs on a path. The total path cost is merged with a stability cost to strike a tradeoff for maximizing data transfers in the entire participatory environment. A polynomial time dynamic programming algorithm is proposed to compute paths of optimum cost.

We propose another novel scheme for Real-time Mobility and Orientation Estimation for Mobile Environments (MOEME), as prediction of user movement paves way for efficient data transfers, resource allocation and event scheduling in participatory environments. MOEME employs the concept of temporal distances and uses logistic regression to make real time estimations about user movement. MOEME relies only on opportunistic message exchange and is fully distributed, scalable, and requires neither a central infrastructure nor Global Positioning System.

Indeed, accurate prediction of contact volume, path capacity and stability and user movement can improve performance of deployments. However, existing schemes for such estimations make use of preconceived patterns or contact time distributions that may not be applicable in uncertain environments. Such patterns may not exist, or are difficult to recognize in soft-real time, in open environments such as parks, malls, or streets.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

Introduction

This dissertation investigates exploitation of opportunistic contacts to enable effective and efficient data transfers in challenged participatory environments. In this chapter, we provide an overview of our research on effective and efficient communication and collaboration in participatory environments. We also introduce participatory environments and provide motivation for the research presented in the rest of the dissertation. Furthermore, we discuss key research problems and their proposed solutions. The chapter is concluded with an outline of rest of the dissertation.

1.1 Participatory environments

Participatory environments are formed when a set of users carrying smart devices come in the vicinity (radio communication range) of each other to execute a common collective task. Such a collection of users may result in the formation of an opportunistic network. There is no end to end connected path between source and destination nodes in opportunistic networks. In order to transfer data or messages between interested users, intermediate nodes may act as ferries by physically carrying information along with them.

Participatory environments and dynamic networks are of considerable interest as they hold promise in offloading traffic from licensed spectrum to unlicensed spectrum, such as Wi-Fi or Bluetooth. However, in order to deploy real-world applications in participatory environment there is a need to study a number of characteristic problems that arise due to their dynamic nature.

1.2 Research problems

In this section, we discuss the research problems that arise in achieving effective communication and collaboration between participating devices in a dynamic participatory environment. As the participants are mobile, often there are no patterns in user movement that can be exploited to achieve reliable and efficient data transfers. Data corruption and under utilization of the network are common problems that arise while deploying applications in participatory environments.

1.2.1 Contact volume

In recent years, the number of smartphone users has exceeded one billion, thus creating the opportunity for developing and deploying useful applications in opportunistic networks. As an example, consider a group of smartphone users running a bandwidth sharing application to download and possibly stream a video. In such an application, a group of nearby users simultaneously download different chunks of a video with the help of a low speed cellular network connection and share it among themselves over a high speed wireless LAN (possibly at no cost) [1]. The advantages of bandwidth sharing include, but are not limited to increased download speeds, better video quality, reliability and reduced delays.

Research problem: In the existing bandwidth sharing schemes [2] [3] [4] [5], it is assumed that the group of users wanting to watch the same video are within proximity of each other and are not mobile. However, situations where a group of mobile users may not be within their radio communication range, but can be connected in an opportunistic network are quite common in such public places as streets, train stations and state fairs. Therefore, it is of paramount importance to have an accurate prediction of contact volume i.e., the maximum amount of data that can be transferred between opportunistically meeting nodes. The prediction of contact volume helps applications to determine what files or data can be reliably transferred to other nodes in the environment.

1.2.2 Capacity and stability of paths

Wireless devices that make up such a network are mobile and resource constrained. Furthermore, the network needs to be self configurable and be able to route data between interested users. In order to tap into the full potential of dynamic networks, researchers have aggressively modeled their structure and properties [6] [7] [8]. Data centric approaches [9] [10], along with temporal reachability [11], for data transfers have also been proposed in the literature.

Research problem: To the best of our knowledge, there is no existing literature on modeling of real-world link capacities along a path in opportunistic networks. In some works link costs are incorrectly assumed to be uniform throughout the network, while in some others, bandwidth limitations are not considered [12]. The individual radios and interfaces may also be different in a set of heterogeneous devices, thus giving rise to greater complexity. Therefore, there is a need to model the real-world path capacities in a participatory environment in order to choose the best path for transferring data between interested nodes. Despite the dynamic nature of such environments, it is important to maintain stable paths in order to minimize overheads due to routing table updates. Moreover, frequent path changes lead to poor data quality and disrupt user experience.

1.2.3 Mobility

As participatory environments rely primarily on user mobility as a mechanism for transporting content and data in general, identifying and modeling user mobility provides researchers and network designers with key insights for improving performance, efficiency and productivity. For example, epidemic routing [13] guarantees shortest latencies in message delivery, however, knowledge of user mobility and diffusion has been exploited by various schemes to considerably cut down energy expended in already resource constrained devices with a small tradeoff on delivery times [14] [15] [16].

Research problem: Though there are several existing works in the literature that aim at modeling human mobility, Self Similar Action Walk (SLAW) and Community-based Mobility Model (HCMM) are the two popular synthetic state of the art mobility models [17] [18]. However, participatory environments are dynamic and there may not be existing patterns, hence there is a need to predict human mobility in real-time to mitigate the effects of randomness in user movement.

1.3 Contributions

To address the aforementioned research challenges we make the following novel contributions towards effective collaboration in participatory environments a reality.

1.3.1 Predicting contact volume between opportunistically meeting nodes

We propose a novel scheme called PCV that effectively predicts the contact volume and enables usability of an opportunistic contact. PCV makes use of the pre-calculated data rate profile expressed as a function of distance, as well as the instantaneous velocity vectors of the mobile nodes. This work makes the following major contributions:

1. Analytical model for estimating contact duration and a scheme for Predicting Contact Volume (PCV) in opportunistic contacts to enhance file transfer capability. The scheme has been shown to work in the absence of user mobility patterns and contact duration distributions.
2. A novel custom Android Application called DatPro: Data Rate Profiling Agent, to learn Bluetooth data rate profiles for outdoor environments. Source code available [19].

3. PCV algorithm predicts the contact volume for pairs of nodes. The algorithm is simple, scalable and runs in linear time in terms of the number of opportunistic contacts a node makes.

1.3.2 COSC: Paths with Combined Optimal Stability and Capacity in Opportunistic Networks

In the aforementioned work, we modeled link costs based on the actual amount of data that can flow between two opportunistically meeting nodes [20]. We define contact volume as the maximum amount of data that can be transferred between two opportunistically meeting nodes. Our studies show that contact volume is dependent on - initial velocities of nodes, distance between nodes, environmental characteristics such as data rate profile and radio characteristics. We model real-world link costs for multi hop paths between source and destination nodes. Main contributions include:

1. Modeling of real world path capacities in opportunistic networks;
2. A scheme to incorporate cost and stability of a path for selecting data transfer paths that guarantee a minimum number of packet transfers;
3. A polynomial time algorithm to calculate desired paths of minimum cost; and
4. Implementation on a test bed of Google Nexus 5, Android 4.4.3 (KitKat) devices to validate theoretical findings.

In order to evaluate our scheme, we run rigorous simulations on both real-world and synthetic mobility traces and discover possible data transfers of up to 120 MB at a communication range of 150 m between the mobile devices. COSC demonstrates 60% reduction in file transfer failures when compared to common methodologies.

1.3.3 Real-time mobility estimation in participatory environments

We present a novel scheme called MOEME: Real-time Mobility and Orientation Estimation of Mobile Environments MOEME. MOEME empowers both users and system architects with the knowledge of user mobility. MOEME estimates relative directional mobilities of all the users in a participatory environment, in addition to counting the number of users present within a desired spatiotemporal radius. We demonstrate how MOEME can be used in a variety of scheduling and resource allocation applications. The accuracy of predicting contact volume can be improved directly with the knowledge of directional mobility of users. Furthermore, it helps in predicting the total volume of information that can be pushed from one end of the network to the other or from a particular source node to a desired destination node.

Major contributions of MOEME include:

1. Estimation of relative directional mobilities of all users in real-time without requiring their movement histories; and
2. Estimation of the number of users, that are likely to be within a spatiotemporal region of interest.

To the best of our knowledge MOEME is the first scheme to make the above mentioned contributions for estimations of user mobilities in mobile environments.

MOEME estimates distance of users and number of users within a distance of 300m with an accuracy of 89%. The estimates of direction of users are 77% accurate for nodes within 200m.

The novelty of MOEME lies in its ability to estimate user mobilities with no *a priori* knowledge of movement histories. MOEME can be employed in indoor as well as outdoor environments. MOEME is fully distributed, lightweight and has a time complexity of $O(n)$ at each node, where n is the number of nodes present in the mobile environment.

1.4 Outline

The rest of the dissertation is organized as follows: We give an overview of the related work and state of the art in Chapter 2. In Chapter 3 we give the details of predicting contact volume between two opportunistically meeting nodes. Chapter 4 extends that idea and models real-world capacities of multi-hop paths and simultaneously considers path stability for efficient collaboration among nodes. Next, chapter 5 gives the details of estimating user mobility in soft real-time. Finally, we conclude in Chapter 6.

CHAPTER 2

Related Literature

In this chapter we present survey of literature related to this dissertation. The survey is organized under the three categories: (1) Contact volume, (2) Data flow in opportunistic networks, and (3) Mobility estimation.

2.1 Contact volume

2.1.1 Finding patterns

There exist several algorithms for predicting future contact times among nodes. The main purpose of these predictions is to enhance message delivery to destinations with smaller bandwidth consumption. Long Vu et al. [21] constructed a model called Jyotish which provides prediction of location, stay duration, and contact of a person altogether with a considerably high accuracy. Marchini et al. [22] developed a mechanism for successfully detecting cyclic movements and predicting the next appearance of the mobile node. As proposed by Yuan at el. [23], nodes themselves determine the probability distribution of future contact times and choose a proper next-hop in order to improve the end-to-end delivery probability.

2.1.2 Throwboxes and RSSI based techniques

Regarding the measurement of contact volumes, Chowdhury et al. [24] proposed a framework based on IEEE 802.11n wireless to give an analytical model on contact volumes. Neto et al. [25] used the findings from [24] and predicts the contact volume as a function of received signal strength indicator (RSSI). Banerjee et al. [26] came up with the idea of using energy efficient throwboxes. These throwboxes are battery-powered stationary nodes with radios and storage, in an opportunistic network setting. These nodes can predict the opportunistic contact volume by using a long range radio equipped with a GPS to detect mobile nodes and to measure their speed and direction.

However, none of the above mechanisms exploit mobility vectors coupled with data profiles at variant distance to optimize contact volumes. Cyclic pattern detection is not feasible for environments with high dynamics. Coming up with a probability distribution function as a predictor of future contact demand sufficient historical information, which is unavailable in many scenarios. . Although being inexpensive, throwboxes can not be deployed everywhere to cover areas where opportunistic networks are formed once in a while. Finally, the idea of calculating of contact volume as a function of RSSI has been proven to be inadequate [27].

2.1.3 Contact volume application

Color Barcode Streaming for Smartphone Systems (COBRA) is an interesting piece of work that utilizes Visual Light Communication (VLC) to transfer information between off the shelf smartphones, using a display screen and on board camera [28]. The foremost contribution is the design of a new color barcode. The researchers have primarily used three more colors i.e., red, green, blue, in addition to the standard black and white, to encode information in a bar code. Special corner blocks and timing references are introduced along with the main code blocks for accelerated extraction and processing at the receiver side.

1. **Corner blocks:** The unique feature of the corner blocks is that the color scheme allows the system to identify the orientation of the bar code in addition to the usual corner tracking.

2. **Timing reference blocks:** Allows direct access to color code sections for faster decoding.

2.1.3.1 Adaptive code generation

Authors consider the fact that, though theoretically smaller code blocks allow to pack in more information, doing so may lead to higher blur, which in turn decreases the system throughput. Therefore, COBRA adaptively adjusts the size of the code block. The idea is that whenever the acceleration experienced by the device is higher, bigger code blocks are used and vice versa.

2.1.3.2 Color ordering and blur assessment

The authors have designed clever techniques to optimize the order of the colors to minimize the blurring effect. Intuitively, most blur occurs at the boundary of different colors, however, placing similar colors adjacent to each other can reduce the blur considerably. Moreover, the concept of blur assessment is employed, wherein each image (the system sends multiple images of the same code) is labeled with a Degree of Blur (DOB) value. The image with the lowest DOB is used for further processing. Apart from this, the authors use HSV values to match the detected color against the most plausible color. This enhances the image, making it more crisp for code extraction.

2.1.3.3 Implementation of COBRA on smartphones

The authors have developed a real-world implementation of COBRA on smartphones and tested its performance against prior work, such as PixNet [29].

2.2 Data flow

One of the earliest works on data transfers in opportunistic networks falls under the category of routing. There are schemes such as single copy [30] and multiple copy [31] that try to improve message delivery ratio while making a trade-off between end-to-end delays. Sadiq et al., presented a method that utilizes nodes' diffusion and proximity to improve both delivery ratio and delays [14]. The fastest method to perform routing in intermittently connected mobile networks is epidemic routing, which entails flooding the message throughout the network [13]. However, this has obvious drawbacks, such as overflowing buffers and significant battery consumption on resource constrained mobile devices. Though routing in essence is related to our work, it generally deals with messages that are short and do not require considerable contact volume.

2.2.1 Data dissemination

Data dissemination has been studied for conventional Mobile Ad Hoc Networks (MANETs) [32] [33]. In general, these systems assume that network paths are rather stable and in some cases generate significant amount of traffic just to maintain knowledge of other nodes' caches. Therefore, they are not suitable for user-based opportunistic networks.

2.2.2 Informed mobile prefetching

Prefetching is a technique where data or instructions are loaded in anticipation of their need in the future [34]. In mobile systems, this hides latency over poor and intermittent wireless connection, but at the same time energy and cellular data usage costs can grow considerably. In this paper the authors propose a scheme called Informed Mobile Prefetching (IMP), that boosts performance of higher level application layer by prefetching data items specifically labeled by an application. The key idea IMP hinges on is an adaptive tradeoff between budgeted resources, such as, battery life and data usage. IMP runs a control loop to check the state of budgeted resources and accordingly adjusts the conversion rate to reduce latency. IMP tries to utilize resources as best as possible, without exceeding the upper bound. The argument is that most schemes will try to minimize the use of budgeted resources, overlooking the fact that data plans do not roll over to the next month. A similar argument applies to battery life, i.e., when a user knows she will recharge the battery by the end of the day, prefetching should be carried out rather aggressively to reduce latency and improve user experience as best as possible.

2.2.2.1 Decision algorithm

There are three parts to the decision algorithm used:

1. **Performance:** The performance benefit of prefetching is calculated as the product of time (T_{fetch}) to fetch an item in the future and the application/class specific accuracy. T_{fetch} is estimated by taking into consideration, the type of network that might be available in future (WiFi or Cellular), its latency and the size of the data item to fetch. Similarly, the application accuracy is estimated using ratio of number of items correctly prefetched to the total number of items prefetched in the past.
2. **Energy consumption:** The total energy benefit of a prefetch is given by $E_{prefetch} - E_{fetch} \times Accuracy$. Where $E_{prefetch} = T_{prefetch} \times (P_{WiFi}\ or\ P_{3G})$. Depending on which network is available and/or cheaper to use currently.
3. **Data usage:** Modeling the data usage cost is easier and intuitive. It is given by $D_{prefetch} - D_{fetch} \times Accuracy$. Where $D_{fetch} = S \times (1 - Availability_{WiFi})$. $Availability_{WiFi}$ is a boolean value, indicating whether the WiFi network is available or not in the future. Similarly, the value of $D_{prefetch}$ depends on whether the WiIFi network is available at the time of prefetch or not.

IMP calculates the net benefit of prefetching by using the above mentioned three quantities. The quantity

$$T_{fetch} \times Accuracy - (cenergy \times (E_{prefetch} - Accuracy \times E_{fetch}) +$$
$$cdata \times (D_{prefetch} - Accuracy \times D_{fetch}))$$

is calculated. If it is positive for a given network, IMP will prefetch it.

2.2.2.2 Implementation

Apart from the decision algorithm for informed mobile prefetching, the authors of the paper have shown a real world implementation of their scheme. They have modified two mobile applications i.e., News reader and K9 email application. They have tested the performance of IMP against other schemes. As mentioned in the paper these schemes are: *"never prefetch anything, prefetch items with size less than an application-specific threshold, prefetch over WiFi when it is available but never over the cellular network, and always prefetch everything."*

2.2.3 Data centric approaches

There are other methods in the literature that take the data centric approach to information dissemination in opportunistic networks. Conti et al., define how a semantic representation of information can be used to determine relevance of information to be exchanged in opportunistic networks [9]. Pietilanen et al., approach the problem of data dissemination from a social networking point of view [10].

2.2.4 Time varying graphs

A closely related work to ours is by Bui Xuan et al., who have proposed algorithms for calculating shortest (in number of hops), fastest and foremost (i.e., earliest arrival) paths in TVGs [35]. All these algorithms are designed to compute the shortest paths to all destinations from a source and a fixed starting time. However, they do not take into account the costs associated with the links, instead, their work focuses on number of hops in calculating the shortest distance. Casteigts et al., study deterministic computations to broadcast data where edges in a TVG do disappear, but appear infinitely often [36]. However, they also only take into account the hop counts.

2.3 Mobility estimation

2.3.1 Routing

In dynamic and pervasive networks, a significant amount of research has focused on efficient routing schemes [37] [38] [39] [40] [41] and content dissemination frameworks [42] [43] that exploit repetitive patterns in human movement. Recently, there has been some focus on content and service distributions in open environments, such as parks, malls etc. where history from past visits is not available [14] [44] or repetitive patterns do not exist at the time scale of few minutes (0 to 15 minutes).

2.3.2 Multi Dimensional Scaling

Multi Dimensional Scaling (MDS) is a powerful centralized technique for node localization [45]. As the scheme is centralized it is quite accurate and performs well. However, as the message over head is high (relaying a lot of messages to a central authority), the required memory, power and computational complexity is high

2.3.3 Two phased localization based on Simulated Annealing

This is a two phased technique, where in the first phase, simulated annealing is used to get the distance estimate of the localizable nodes. In the second phase, the error caused by flip ambiguity is removed. This is a centralized approach, which means the communication overhead and computation costs are high, but the accuracy of localization is good.

The technique is most suitable in an environment where the nodes are well connected and resilient to failures. Moreover, it is important that the central agent, handling all computations, is powerful and robust.

2.3.4 RSSI based centralized localization

This is a three step process, where in the first step a general map of the network is formed. In the next step, anchor nodes are used to get a view of distance ranges between the nodes. Finally, an optimization problem is solved to get an accurate position of the nodes. The method is both power hungry and complex because of the optimization involved. In terms of implementation, the memory and networking costs are high as well.

2.3.5 Coordinate System Stitching

There are two main methods that fall under this category of localization.

1. **Construction of Global Coordinate System in a network of Static Computational Nodes from Inter Node Distance:** In this approach a spatial map and the distance matrix are computed for the whole wireless network. Once this step is completed, discrepancies in the aforementioned data are minimized using Euclidean translations, rotations and reflections. The scheme has good accuracy without the need of explicit anchor or seed nodes. However, there is high communication cost associated with this method. The primary reason being the transfer of coordinate system information from the source to every other node in the network. For similar reasons, the addition of each node to the network adds complexity to the scheme making it less scalable.
2. **Cluster based approach:** This technique has good accuracy and low communication overhead. The main advantage of the scheme is deployment for an application where the nodes are inserted dynamically. However, if the graph is poorly connected, which

can be a result of a difficult terrain or inaccurate sensors (with high noise variance), then the cluster based approach may not be able to localize a considerable number of nodes.

2.3.6 Interferometric Ranging Based Localization

This is a localization approach that requires no other sensors, other than the ones used for wireless radio communication [46]. However, the problem of localization itself is NP-complete. As pointed out in the paper, there are genetic algorithms and other RSSI based techniques that optimize the solution by reducing the search space. Though the accuracy of these methods is high, but at the same time the messaging overhead is quite large. The overall advantage of Interferometric Ranging Based Localization is that the measurements can be very precise, however, the scheme requires large sets of measurements, which makes it less suitable for smaller WSNs.

2.3.7 Beacon based distributed localization

Beacon based distributed localization techniques can be broadly categorized into three methods [45].

1. **Diffusion based:** For the diffusion based method the position of the node is estimated by calculating the centroid of the neighboring already localized nodes. As the scheme requires a large number of nodes, the overall power consumption can get considerably large, increasing the computation and memory requirements on the whole. However, the scheme is robust to failures, as there are many nodes and if some of them fail, (though not the critical ones) localization with good accuracy can still be achieved. The strong points of this technique are simple and easy implementation. As mentioned before, the scheme will work best in an environment, where deployment of a large number of nodes can be achieved easily.

2. **Bounding box:** As the name suggests, these techniques form a bounding box around and node and subsequently, redefine its position based on this view. The main advantage of this technique is good accuracy which is achieved by using beacons or anchor nodes that are located on multi hop paths. The two of the algorithms discussed in the paper have a provision of making a tradeoff between communication and computation costs.
3. **Gradient based:** In the gradient based approach the nodes keep counters to determine the hop count from the 'seed' nodes. Seeds are similar to anchor nodes, which are aware of their global coordinates. The advantage of gradient based approach is that it is scalable in terms of addition of both seeds and regular nodes. Moreover, it is resilient to node failures. However, the accuracy of the system depends on the node density in the environment, where more number of nodes translate into higher localization accuracy. Finally this scheme is easy to deploy but works best in terrains where there are no obstacles blocking the communication. Such obstacles will lead to incorrect hop counts.

2.3.8 Plausible mobility

There is only one work [47] that aims to create plausible mobility merely through user contacts but requires a centralized server to keep track of all contacts in real time. In contrast, we present the first work that estimates: 1) relative directional mobilities of all users in real-time without requiring their movement histories; and 2) the number of users, that are likely to be within a spatiotemporal region of interest, through opportunistic contacts in a completely decentralized manner.

CHAPTER 3

PCV: Predicting Contact Volume for Reliable and Efficient Data Transfers in Opportunistic Contacts

3.1 Introduction

There are various challenges posed by participatory environments for delivering content and transferring data between participating nodes. In this chapter we identify and deal with one such challenge. We have focused our attention on two mobile participating nodes, and leverage their physical properties to reliably and efficiently transfer data between them. The work presented in this chapter lays the groundwork for extending the scheme to multiple nodes in the environment.

Employing PCV[1], it is possible to exploit contact durations among pairs of intermediate nodes to deliver video chunks to a group of mobile users reliably. When the contact duration is short, it is worthwhile to transfer at least part of the information, as compared to the full video chunk a node may have. The latter has a higher chance of failure than the former in opportunistic networks. In such a case, an accurate prediction of contact volume, defined as the maximum possible amount of data transferable during a contact, can help in making better choices (in terms of cost, delay, and/or fidelity) at the application level. Knowledge of contact volume at the start of an opportunistic contact would facilitate efficient management of resources in challenging network environments to meet the needs of a variety of application and user requirements. It should be noted that the user mobility patterns are haphazard in dynamic and unpredictable public environments. Existing works

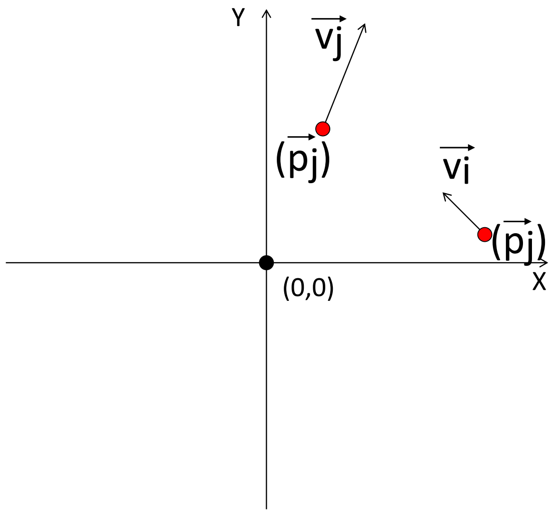
[1]The work presented in the chapter was completed during 2012-13, when the PhD candidate and the advisor were affiliated to the University of Texas at Arlington.

utilizing cyclic mobility patterns or forecast peer interactions [48] [22] for effective information dissemination in opportunistic networks, are likely to perform poorly in such an environment. Furthermore, the use of contact duration distribution [49] for the prediction of contact volume does not hold promise for similar reasons.

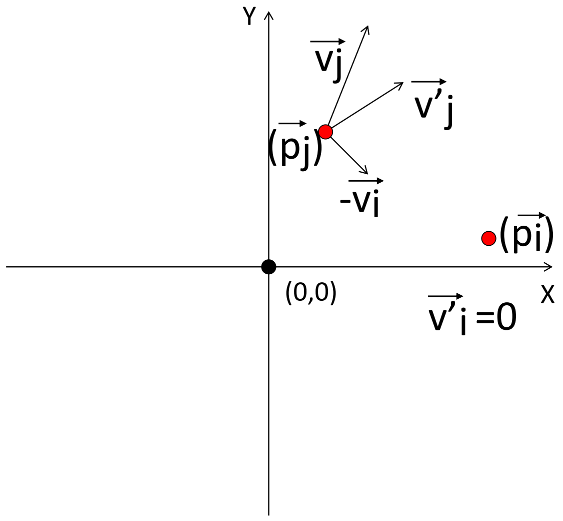
In this chapter, we propose a novel scheme PCV that is used to predict the contact volume between two opportunistically meeting nodes based on their initial position and velocity vectors at the time of contact. The energy consumption is reduced by PCV by switching on Global Position System on a device only at the time of contact. PCV is the first scheme that also supports bandwidth sharing among mobile users who may or may not be within each other's communication range.

PCV is primarily developed to assist applications in opportunistic networks, by making data transfers reliable and efficient. Conceptually, PCV serves as a middleware interface between the application and the opportunistic network layers. PCV estimates contact volume for any pair of nodes meeting opportunistically in soft real-time and subsequently sends this information to the application layer above. Our major contributions are summarized below.

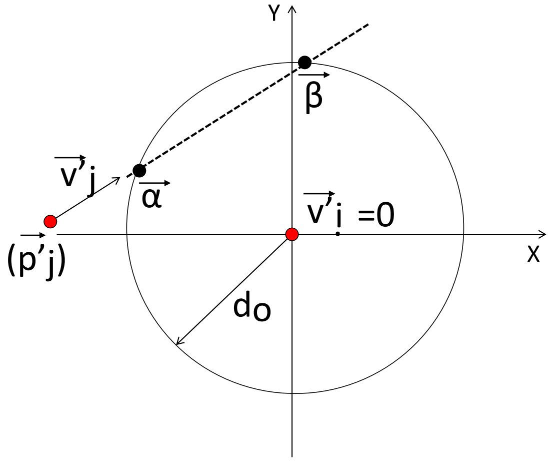
- **System Model:** Analytical model to compute instantaneous distance and contact duration between nodes. These values are used in the PCV algorithm to compute contact duration and instantaneous distance between the nodes in an opportunistic network.
- **Data Rate Profile:** A custom Android Application to acquire data rate profiles of contacts between nodes in an opportunistic network to facilitate the prediction process. Data rate profile incorporates wireless characteristics for different environments. In order to validate PCV, we have collected data rate profiles for a real world scenario, wherein the users carried Android smartphones.



(a) Initial scenario, i.e., before applying transformations



(b) In the Inertial Frame of Reference of node n_i



(c) After applying Euclidean Translation to position n_i at (0,0)

Figure 3.1: Series of transformations applied for calculating contact duration of two nodes

- **PCV Algorithm:** PCV algorithm predicts the contact volume for pairs of nodes. The algorithm is simple, scalable and runs in linear time in terms of the number of opportunistic contacts a node makes.

The primary focus of PCV is to make file transfers reliable and efficient to improve the usable bandwidth of the underlying opportunistic network. However, we do realize the privacy and security issues that may arise out of sharing data among untrusted nodes. To address this, PCV is designed to have a compartmentalized structure in order to accommodate anonymity and encryption techniques. Moreover, PCV can also use any state of the art routing and forwarding schemes for opportunistic networks [14].

3.2 System Model

In this section, we develop a model for opportunistic contacts between two nodes in an opportunistic environment. Expressions for contact duration and instantaneous distances between nodes are derived using nodes' positions and velocities at the time of contact. Both contact duration and instantaneous distances of the nodes capture the dynamics of an opportunistic contact, and are critical for the estimation of contact volume.

3.2.1 Contact Duration

Let $\mathbf{N} = \{n_1, n_2, n_3, ..., n_l\}$ be the set of l nodes, where all nodes have a homogenous radio communication range d_0. Whenever two nodes n_i, $n_j \in \mathbf{N}$, where $1 \leq i, j \leq l$, come within each others' radio communication range, they are said to be in 'contact'. Let $t_{ij} = t_{ji}$ be the duration of this contact. For the sake of simplicity we assume the nodes to move in a two dimensional space. However, the analysis can easily be scaled to three dimensional space. Instantaneous position and velocity vectors are used to predict the contact duration. Let

$$\vec{p_i} = \begin{bmatrix} p_{ix} \\ p_{iy} \end{bmatrix}, \vec{p_j} = \begin{bmatrix} p_{jx} \\ p_{jy} \end{bmatrix} \quad \vec{v_i} = \begin{bmatrix} v_{ix} \\ v_{iy} \end{bmatrix}, \vec{v_j} = \begin{bmatrix} v_{jx} \\ v_{jy} \end{bmatrix}$$

be the instantaneous position and velocity vectors of the nodes n_i and n_j, at the start of the contact.

3.2.1.1 Inertial Frame of Reference and Euclidean Translation

In node n_i's inertial frame of reference, applying Euclidean Translation to position n_i at the origin, the transformed position and velocity vectors of n_i and n_j are,

$$\vec{p_i'} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} , \vec{p_j'} = \begin{bmatrix} p_{jx} - p_{ix} \\ p_{jy} - p_{iy} \end{bmatrix} \quad \vec{v_i'} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} , \vec{v_j'} = \begin{bmatrix} v_{jx} - v_{ix} \\ v_{jy} - v_{iy} \end{bmatrix}$$

respectively. Fig. 3.1 illustrates the effect of these transformations in succession. The duration t_{ij} is the time for which n_j stays within distance d_0 of node n_i, which is essentially the time it takes for a node to traverse a chord in a circle of radius d_0, as depicted in Fig.3.1c. The chord through the circle, defined by $x^2 + y^2 = d_0$ is traced by a line $y = mx + \sigma$, where

$$m = \frac{v_{jy} - v_{iy}}{v_{jx} - v_{ix}}$$
$$\sigma = (p_{jx} - p_{ix}) - (\frac{v_{jy} - v_{iy}}{v_{jx} - v_{ix}} \times p_{jx} - p_{ix})$$

Solving the equation of the line and circle simultaneously leads to a quadratic equation in x, as follows:

$$(m+1)x^2 + 2mcx + \sigma^2 - {d_0}^2 = 0 \tag{3.1}$$

Let the solutions of the quadratic Equation (3.1) be

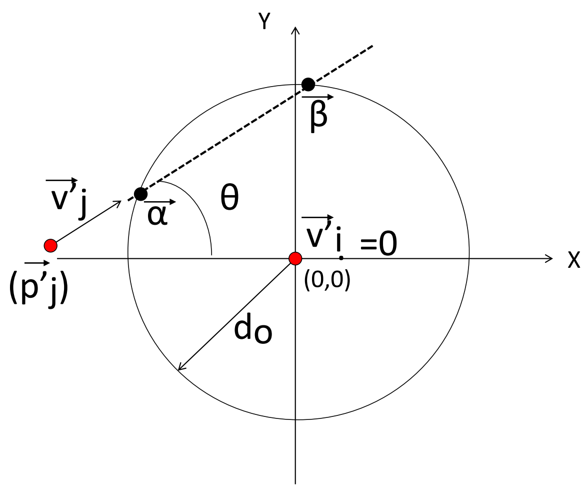
$$\vec{\alpha} = \begin{bmatrix} \alpha_x \\ \alpha_y \end{bmatrix} \quad and \quad \vec{\beta} = \begin{bmatrix} \beta_x \\ \beta_y \end{bmatrix}$$

The effective displacement of chord that node n_j traverses while staying in contact with n_i is expressed as

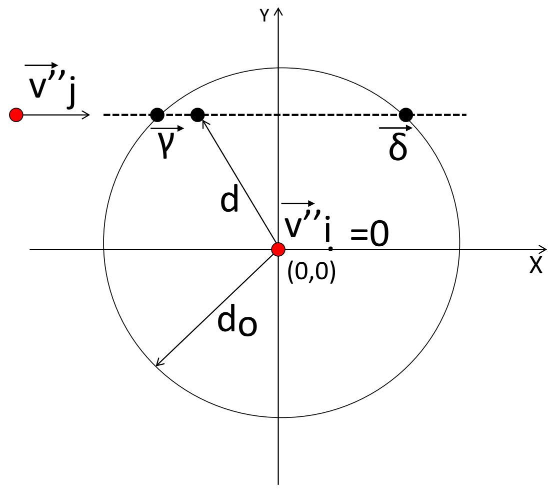
$$\vec{\lambda} = \vec{\beta} - \vec{\alpha} \tag{3.2}$$

Therefore, the final contact duration of the nodes is given by

$$t_{ij} = \frac{\lambda_x}{v'_{jx}} = \frac{\lambda_y}{v'_{jy}} \tag{3.3}$$



(a) Before


Y
X
$\overrightarrow{v''j}$
$\overrightarrow{\gamma}$
$\overrightarrow{\delta}$
d
$\overrightarrow{v''i}$ =0
(0,0)
do

(b) After

Figure 3.2: Effect of rotational transformation by angle θ

3.2.2 Instantaneous Distance

In order to make an estimate of the contact volume, the instantaneous distance between nodes n_i and n_j is calculated as a function of time. This will be used in Section 3.3 to express the data rate profile as a function of time. In wireless communications, the received signal power P_{rec} at the receiver varies inversely with the distance d from the sender. The relationship is concretely represented as $P_{rec} \propto 1/d^{\psi}$, where the exponent ψ depends on the environment in which the nodes operate. It is also known that the transmission data rate depends on P_{rec}, making the data rate a function of instantaneous distance [50]. In a dynamic environment the nodes are usually mobile during an opportunistic contact. Assuming a constant data rate between these mobile nodes for the entire duration of contact is very likely to lead to incorrect estimations of contact volume. Therefore, our proposed model accounts for the changing distance and hence, variable data rate during the contact period.

3.2.2.1 Distance as a function of time

In Section 3.2.1 we see that the essential dynamics of the model are characterized by $\vec{v_i'}$, $\vec{v_j'}$, $\vec{\alpha}$ and $\vec{\beta}$. In order to exploit the inherent symmetry in the problem, a rotational transformation is applied as follows:

$$\begin{aligned}\mathbf{M}' &= \begin{bmatrix} v''_{ix} & v''_{jx} & \gamma_x & \delta_x \\ v''_{iy} & v''_{jy} & \gamma_y & \delta_y \end{bmatrix} \\ &= \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} v'_{ix} & v'_{jx} & \alpha_x & \beta_x \\ v'_{iy} & v'_{jy} & \alpha_y & \beta_y \end{bmatrix} \\ &= \mathbf{RM}\end{aligned}$$

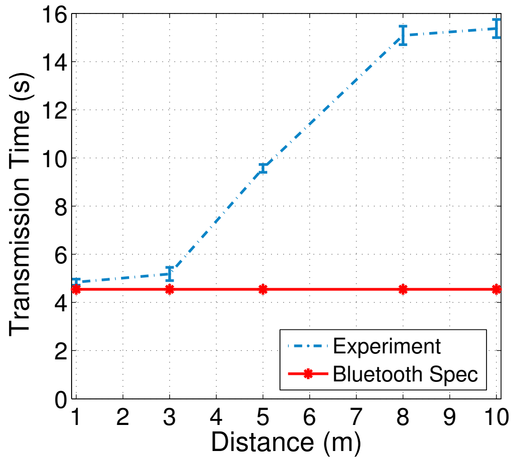
where, $\theta = arctan(\frac{v'_{jy}}{v'_{jx}})$. This rotational transformation moves every vector in $\mathbf{M}$ by the angle θ clockwise about the origin to give their rotated counterparts in $\mathbf{M}'$. Fig. 3.2 shows the result of this rotational transformation. Note that after rotation, the trajectory of n_j is parallel to the horizontal axis, therefore:

$$\gamma_y = \delta_y; \;\; v''_{jy} = 0; \;\; \vec{v''_i} = \vec{v'_i} = 0. \tag{3.4}$$

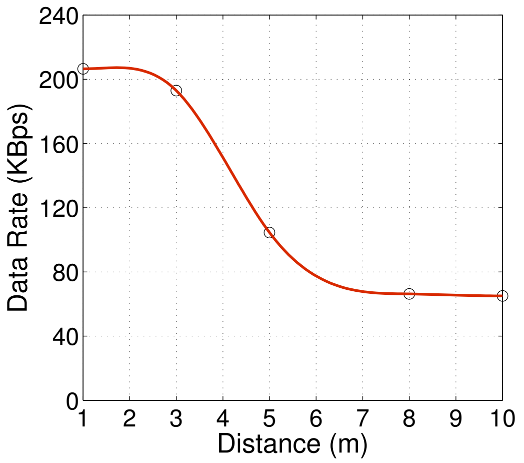
Using (3.4), we can trace the trajectory of node n_j after it comes within the communication range of n_i. Thus, for any time t where, $0 \leq t \leq t_0$ (contact duration), the motion of n_j in two dimensional space is given by

$$x = \gamma_x + (v''_{jx} \times t) \tag{3.5}$$

$$y = \gamma_y \tag{3.6}$$



(a) File Transfer time at different distances between two Nexus One smartphones (*filesize* =1000 KB)



(b) Cubic spline for obtaining data transmission rate at different distances

Figure 3.3: DatPro results

From Equations (3.5) and (3.6), the instantaneous distance between n_i and n_j is given by,

$$d = \sqrt{(\gamma_x + (v''_{jx} \times t))^2 + \gamma_y^2} \tag{3.7}$$

3.3 Data Rate Profile

The data rate between wirelessly communicating nodes depends on the distance between the nodes and the environment in which they operate. Specifically, the environment characterizes the amount of interference faced by the communicating nodes. For example, environments cluttered with buildings, trees or terrain irregularities will give rise to a higher value of the exponent ψ, i.e., the received power at the receiver drops steeply with the distance from the sender, thereby lowering the data rate at a given distance $d \leq d_0$. It is usual practice to determine the value of ψ empirically for a given environment. Empirical models have the benefit of limited reliance on detailed knowledge of the terrain and speed of execution [51]. We have developed an Android Application that learns the data rate profile for a given environment. Subsequently, in order to interpolate the discrete set of data rates obtained from the application, cubic splines [52] are used.

3.3.1 Custom Android Application for Learning Data Rates

We developed a custom Android Application called DatPro in order to obtain data rate profiles as a function of distance. DatPro is primarily designed to obtain Bluetooth data rates in a real-world scenario. However, it can be enhanced to work with 802.11n and WiFi Direct [53]. DatPro serves as a proof of concept and aids the application designers to better understand the data rate constraints prevalent in a specific opportunistic network environment.

In this chapter, the DatPro is used to collect data rate profile in the outdoor environment of our university campus. The experiments were carried out in the walk-ways beside a multi-storied building and in a parking lot, of the university. Table 3.1 shows the different wireless devices used for collecting data rate profiles. As all the devices had similar trends in data profiling, we use results obtained only from Google Nexus One in the following sections. This particular device ran Android version 2.3.6 operating system and Bluetooth version 2.1 EDR (Enhanced Data Rate). Fig. 3.3a shows the effect of distance on the file transmission period. It is evident that with increasing distance, a file requires more time for transmission, implying the drop in data transmission rate with distance.

Table 3.1: Wireless devices used to gather data rate profiles

Device Name	Platform
Google Nexus	Android 2.3.6
Motorola Milestone	Android 2.1
Acer Iconia A500	Android 4.0
Nook Color	Android 2.3.7

3.3.2 Cubic Splines for Estimating Data Rates

Once the data rates are measured at different distances of separation between the wirelessly communicating devices, a curve is fitted through these discrete set of points to compute the data rate function denoted by $R(d)$. Cubic splines are used for curve fitting, as they have the advantage of giving good estimates without the danger of over fitting. Moreover, as the first and second derivatives are continuous on the break points, the curve is nicely smooth [52]. Fitting the data rate points with cubic splines defines the data rate profile as,

$$R(d) = \begin{cases} R_1(d) & d_1 \leq d \leq d_2 \\ R_2(d) & d_2 \leq d \leq d_3 \\ . & \\ . & \\ . & \\ R_k(d) & d_k \leq d \leq d_k + 1 \end{cases}$$

where the kth cubic polynomial is represented as

$$R_k(d) = a_k(d - d_k)^3 + b_k(d - d_k)^2 \\ + c_k(d - d_k) + d_k$$

and $d_1, d_2, ..., d_k$ are the distance values at the break points. Using cubic spline on the set of data presented in Fig. 3.3a, we plot the changing data rate against distance in Fig. 3.3b. This data rate profile is used to measure the final contact volume possible when node n_i meets n_j. It is also noted that the actual data rates obtained in this environment are considerably lower ($R \leq 220KBps$) than the ones mentioned in the official Bluetooth 2.1 EDR (Enhanced Data Rate) specifications.

Algorithm 1 Contact volume estimation by PCV

```
Require: $R(d)$,$\vec{v}_i$, $\vec{v}_j$ , $\vec{p}_i$ and $\vec{p}_j$
  for all Opportunistic contacts do
    Transform velocity and position vectors to $\vec{v}'_i$, $\vec{v}'_j$, $\vec{p}'_i$ and $\vec{p}'_j$
    $t_{ij}$ ←Evaluate eq. (3.1), (3.2) and (3.3)
    $d \leftarrow$ eq. (3.7)
    Substitute $d$ in $R(d)$ to get $R(t)$
    $V'_c \leftarrow$ Evaluate eq. (4.1)
    return $V'_c$
  end for
```

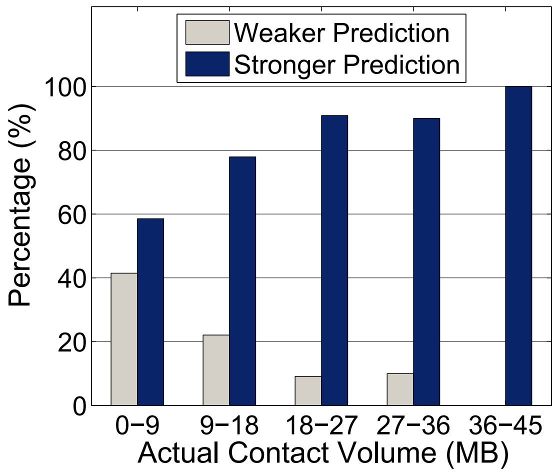
3.4 Contact Volume

In Sections 3.2 and 3.3 we have derived the important parameters that PCV uses to estimate the contact volume, defined as, the maximum amount of transferrable data during an opportunistic contact. This section glues together contact duration, instantaneous distances and data rate profile in an algorithm that forms the backbone of PCV.

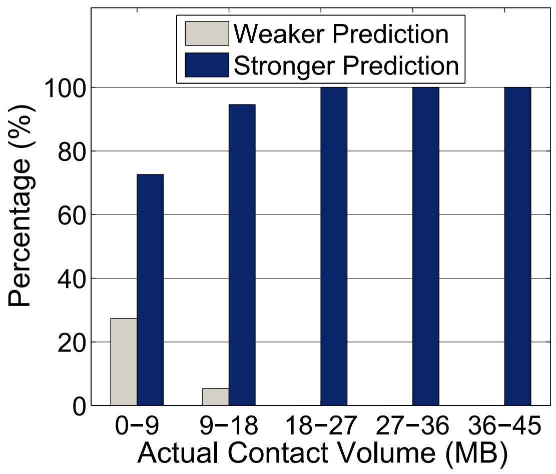
Let dV_c be the small amount of data that can be transferred during a contact. Then the instantaneous data rate is represented as, $\frac{dV_c}{dt} = R(t)$ where dt is an infinitesimal time unit. To derive an upper bound on the total amount of data that can be transferred, i.e., the contact volume V'_c, the above expression is integrated over the contact duration t_0.

$$V'_c = \int_0^{t_0} R(t)\, dt. \tag{3.8}$$

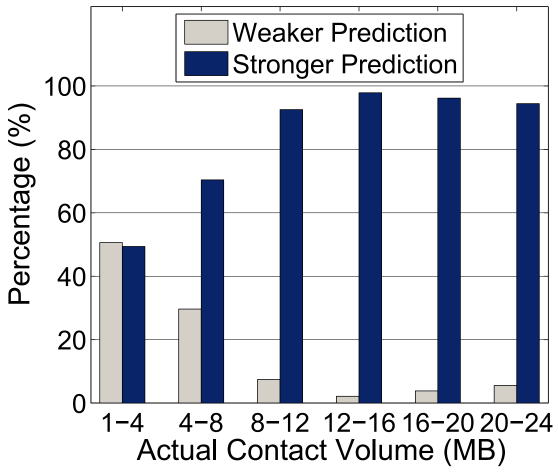
PCV estimates the contact volume at the start of an opportunistic contact as detailed in Algorithm 1.



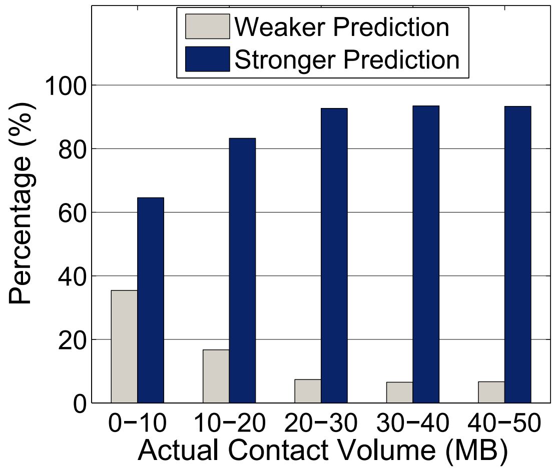
(a) Disney World



(b) State Fair



(c) TLW



(d) SLAW

Figure 3.4: Prediction strength for different ranges of actual contact volume

3.5 Performance Evaluation

In this section, we present results of exhaustive simulation experiments pertaining to both real world mobility traces [54, 55] as well as synthetic mobility traces [17, 56]. Data profiles from DatPro are used to predict the contact volume.

3.5.1 Simulation Setup

The real world traces used for simulation were acquired from two groups of volunteers who visited the NC State Fair [54] and Disney Land in Orlando, [55] respectively, where each of the volunteer's positions were recorded at 30 seconds intervals. There are 19 and 41 log files for the State Fair and Disney Land traces respectively. We consider each file as a mobility trace of one independent smartphone bearer agent or node. The State Fair and Disney Land traces span over a period of less than 3 hours and less than 12 hours, respectively.

Additionally, we use two synthetic traces based on the Truncated Levy Walk (TLW) [56] and Self-Similar Least Action Walk (SLAW) [17], to emulate the statistical features observed in human mobility.

An opportunistic *contact* begins when the distance between the nodes is $\leq d_0$ and ends when the distance is $>d_0$. The contact volume calculated through PCV is called *predicted contact volume*. We also define the *state* of a node, as its velocity at the beginning of a contact.

In our simulations, during each opportunistic contact, the objective is to transmit a file. At the instance when two agents come in contact of each other, the PCV algorithm makes a prediction of the contact volume. If the file size (S_F) is less than the predicted contact volume (V_c), then PCV initiates transmission. Otherwise the transmission is delayed. If the predicted volume is in the range of 70%-100% of the actual contact volume, we term the prediction as *Stronger Prediction*. Otherwise, we call it a *Weaker Prediction*. The choice of range for differentiating between stronger and weaker prediction is justified in Table 3.3, which demonstrates the results obtained from all the test scenarios. The range is chosen carefully based on the minimization of the false negatives. Through experimentation, we observed that the percentage of false negatives falls with decrease in PCVR range as illustrated in Table 3.3. For example, for Disney World the percentage of false negatives

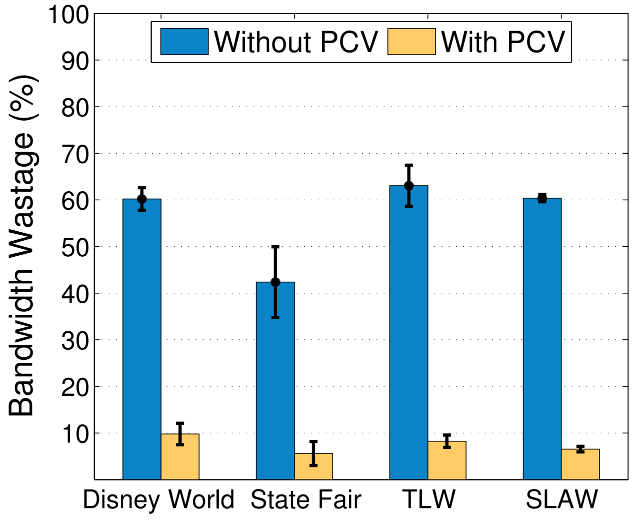


Figure 3.5: Bandwidth wastage with and without prediction.

drops from 7% at a PCVR range of 50% to 100%, to 5% at a range of 70% to 100%, and remains at 5% for the range of 80% to 100%. As this trend is consistent across all traces, we have chosen 70% to 100% as a safe choice for stronger prediction. It should be noted that a weaker prediction is not synonymous with a wrong prediction, which we elaborate in section 3.5.2.

Simulation details and Relevant Metrics: We developed a custom simulator in MATLAB. To retrieve each node's position every second in real world traces, we use linear interpolation between two sample points from the trace, which are temporally 30 seconds apart.

We generate synthetic traces of 50 persons for 6 hours, using TLW [56] by modifying the implemention suggested in [57]. We set the Levy exponent for flight length distribution as α_l = 1.6 and Levy exponent for pause time distribution as β_l = 0.8. The choice of α_l and β_l is also inspired from [57], where the authors successfully generated the inter-contact time distribution of the UCSD trace [58]. The simulation area is chosen as 1 km x 1 km, and the pause times are kept within the range of 30 seconds to 5 minutes. Position information

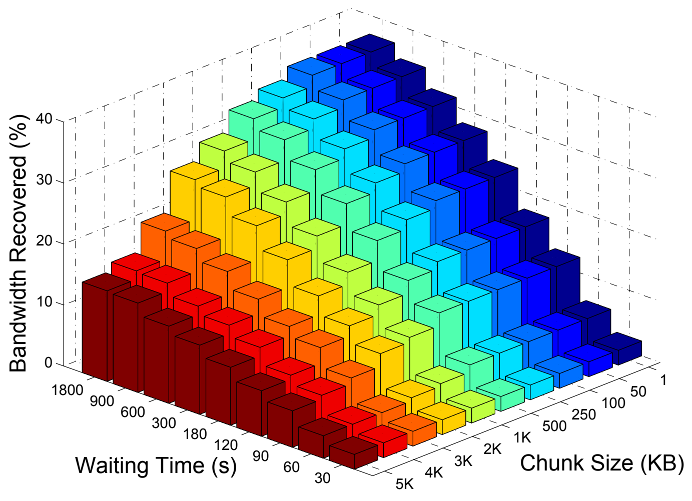


Figure 3.6: Effects of chunk sizes and waiting times on bandwidth recovery for Disney World trace

is logged after every 10 seconds. We generate the intermediate poisitions between two sample positions using linear interpolation. In addition, synthetic mobility traces of 50

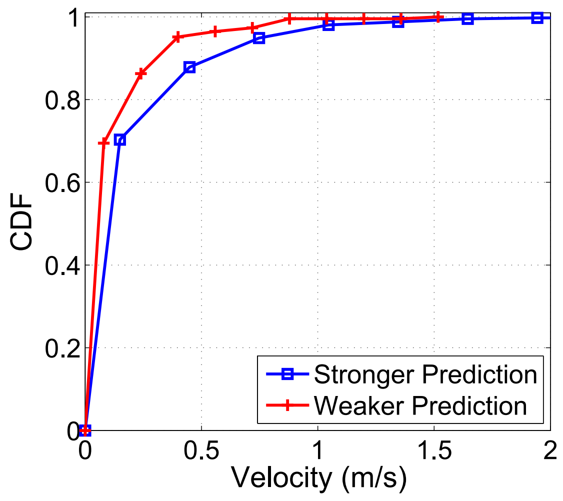
Table 3.2: Distribution of files transferred

File Type	Percentage among all files	(mean, standard deviation) in megabytes
Image	48	(0.5,0.4)
Text	35	(0.75,0.6)
Video	3	(15,12)
Audio	3	(6.3,5.5)
Others	11	(2.3,2)

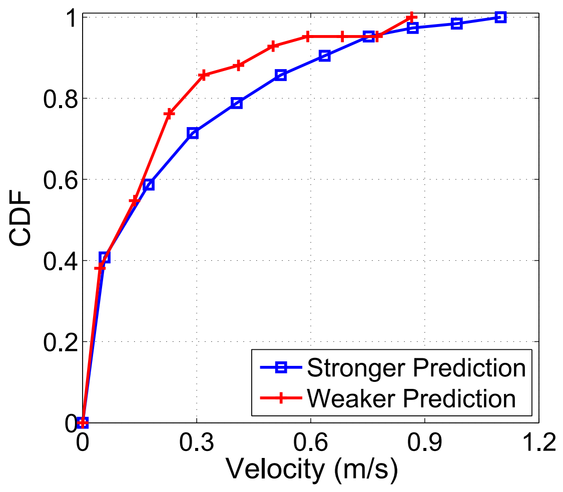
Table 3.3: Percentage of false negatives at different estimated contact volume ranges [Contact Volume Ratio (Predicted vs Actual) = PCVR]

PCVR	Disney World	State Fair	TLW	SLAW
$0.5 - 1.0$	7%	9%	7%	6%
$0.6 - 1.0$	6%	7%	6%	5%
$0.7 - 1.0$	5%	7%	5%	4%
$0.8 - 1.0$	5%	7%	5%	4%

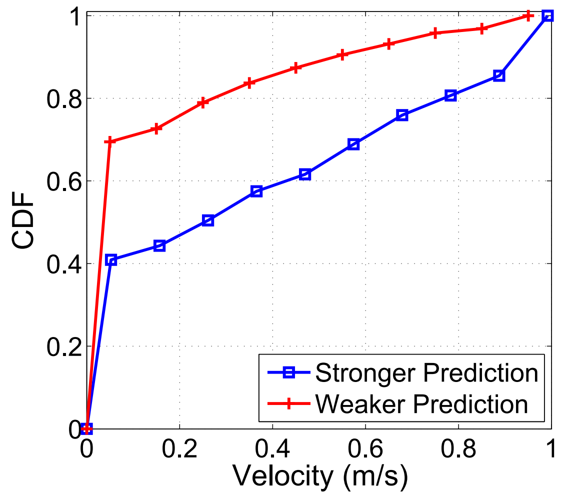
people for 6 hours, are generated using SLAW [17], also by modifying the implementation suggested in [59]. Thus 600 waypoints are generated within a square area of side length of 1500m. The distance parameter α_s is set to 3, Levy exponent for pause time β_s is set to 1, Hurst parameter for waypoints (H) is set to 0.75. A node's pause time varies from 10 seconds to 5 minutes. We record each node's position at 30 second interval and retrieve the intermediate positions through linear interpolation.



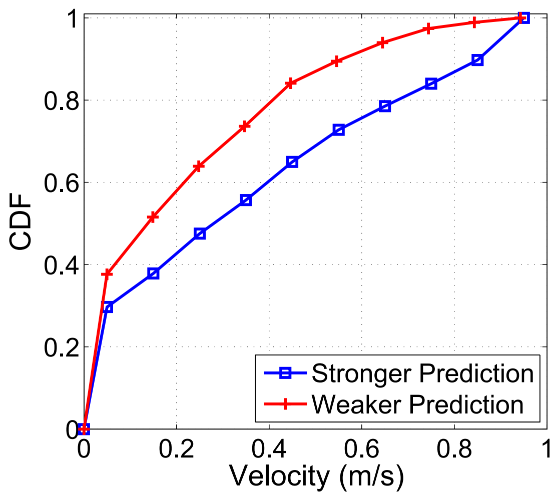
(a) Disney World



(b) State Fair



(c) TLW



(d) SLAW

Figure 3.7: Cumulative distributive function (CDF) of stronger and weaker predictions against nodes' initial state

In order to determine the file sizes, to be transferred in the simulation, we consider PCV's prospect in potential human centric applications aimed at online social network (OSN) and Pocket Switched Network (PSN). This makes the distribution of the downloaded multimedia contents by smartphones as a feasible candidate for determinining the sizes of our synthetic files.

To this end, results from [60] and [61] are used for assessing the downloaded contents both in terms of the number of files and bytes and the average file size. In addition, using the results from [62], we use log-normal distribution with separate parameters for each type of files. The combined decisions are elaborated in Table 3.2.

The following performance metrics are considered in the simulation studies:

- *Prediction Strength*: Reliablity and success potential of the predictions with PCV, for different ranges of actual contact volume.
- *Bandwidth Saving*: The range of possible bandwidth savings through the implementation of PCV.
- *File Chunk size and Waiting Time Frame*: Impact of variable chunk sizes and variable waiting time frame, on bandwidth savings, after initial file transmission distruption.
- *Initial State*: Influence of nodes' initial state, defined by their velocity, on the prediction accuracy.
- *False Negative*: Impact of false negative predictions in PCV.

Table 3.4: Percentage of maximum possible data transmission in real experiments (compared to Bluetooth Specification)

Disney World	State Fair	TLW	SLAW
36.4%	41.9%	32.8%	37%

3.5.2 Simulation Results

The maximum amount of transferable data during a contact volume is computed using the DatPro results plotted in Fig. 3.3b. Table 3.4 shows the comparison between the practical higher limit of total data volume calculated through DatPro and theoretical higher limit mentioned in Bluetooth specifications. However, without a sophisticated prediction mechanism, even the practical higher limit is difficult to achieve.

Fig. 3.4 shows the percentage of stronger and weaker predictions for different ranges of actual contact volumes. As evident from the charts, the predictions are stronger for larger contact volumes. This gives an insight about the PCV's effectiveness in reducing bandwidth wastage, as the system suffers most in terms of energy and bandwidth due to its failure to transmit larger files. Again, recall that a weaker prediction does not imply a wrong prediction. Both over-prediction and under-prediction may eventually lead to successful transfer of many files. However, over-prediction runs the risk of initiating the transfer of a file larger than the actual contact volume, which is destined to fail, thereby resulting in bandwidth wastage. On the other hand, under-prediction may lead to disallowance of the initiation of eligible file transfers, resulting in unused resource. In our experiments, the average percetage of successful transmission in case of weaker predictions are 67.3%, 62.1%, 68% and 73% in terms of data volume for Disney World, State Fair, TLW, and SLAW respectively.

Fig. 3.5 compares bandwidth wastage in two cases: with PCV and without PCV. From the graph, it is evident that over 50% of bandwidth is wasted when nodes try to transmit during oppportunistic contacts, without predicting. However, if they make decisions based on PCV, the wastage is reduced to less than 10%. This is demonstrates a significant achievement of PCV.

Transmitting data in chunks can further reduce the bandwidth wastage. It can be coupled with a waiting time, which means that the transmitting node will wait for a specific time frame to re-establish contact with the recieving node, after the initial contact is distrupted. Trying to transmit the whole file together, results in wastage of bandwidth. In Fig. 3.6 we plot the percentage of bandwidth, recovered from this wasted bandwidth, against different chunk sizes and waiting times in Disney World scenario[2]. Since more files transmit successfully with smaller chunks and larger waiting times, the bandwidth wastage is reduced. On the other hand, very small chunk sizes incur additional overhead whereas very large waiting times increases buffer consumption. Considering this trade-off, the applications can decide on the best operating point for a suitable chunk size and waiting time frame.

From Fig. 3.7 it is evident that the state of the node, which is characterized by its velocity, also affects PCV performance. Velocities are plotted along the x axis and their respective cumulative distribution function (CDF) is plotted along the y axis. As we can see from the graph, for weaker predictions, the CDF grows much faster at the beginning, meaning that when the nodes have relatively smaller velocity, it is difficult to predict their movement, which is intuitive. For example, if both nodes remain static, the predicted contact volume can be extremely large, which may not be true in reallity.

One major concern about any prediction scheme can be the false negative generation rate, that blocks the otherwise eligible transmissions. In a worst case scenario, the available bandwidth may remain unutilised, thus offsetting the bandwidth savings. To quantify this aspect, we plotted the percentage of false negatives with respect to the number of files as well as the unused bandwidth. As illustrated in Fig. 3.8, the portion of the files affected by false negatives is quite small to have an impact on the overall gain of the system. More importantly, in terms of bandwidth less than 10% , implying that smaller files are more likely to be affected.

[2]Results for Disney World, TLW, and SLAW are similar to State Fair scenario, so we do not report them due to lack of space

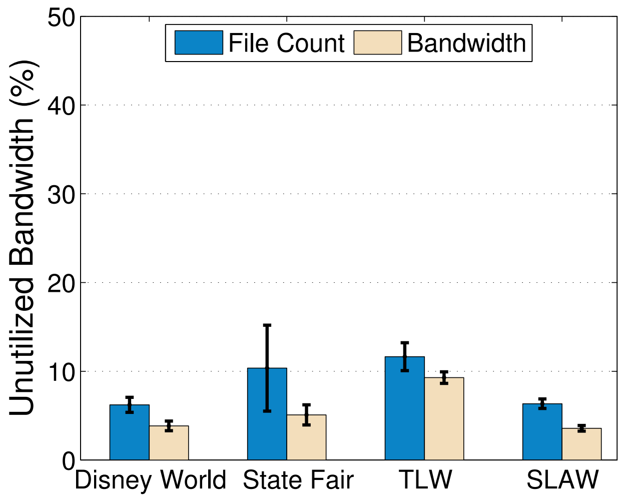


Figure 3.8: Effect of false negatives

3.6 Discussion

3.6.1 Conserving Battery Power

GPS and accelerometer readings on contemporary smart phones give a measure of position and instantaneous velocity. Using accelerometer readings instead of GPS has been shown to consume less energy [63] [64]. Moreover, as accelerometers are usually active on smartphones to detect screen orientations, their usage does not incur additional cost.

In order to minimize battery consumption, PCV, senses a node's position and velocity only at the time of contact. At the start of a contact, to detect whether a user is walking versus other types of movement, such as when the user is stationary but moving the device, techniques like TransitGenie [65] can be used. Furthermore, once it is known that the user is actually on the move at the start of the contact, schemes like Triggered Sensing [66] can be put to use. Here a low-power sensor e.g. accelerometer that detects the movement of the user, can trigger the GPS to gather velocity and position measurements. PCV then uses this information to make a prediction of the contact duration.

3.6.2 Bandwidth Sharing Application

From the simulation experiments, it is observed that single opportunistic contacts in most cases are inadequate to transfer considerably large volume files. As a result, users have to rely on the cellular networks to download files of large volume. However, wireless data plans are still very expensive, which makes the large data bandwidth consumption unaffordable for many. Experimental results illustrated in Fig. 3.6 demonstrate that large files can be successfully transferred in multiple small chunks.

For the bandwidth sharing scenario depicted in Section 3.1, a set of mobile nodes interested in a particular large file, first download separate chunks of the file on the high cost cellular network and then employ distributed sharing techniques [1] to procure the remaining chunks, on the low (or no) cost wireless network. PCV can further augment the distributed sharing process by fine tuning chunk sizes to available contact volume and efficient usage of available bandwidth.

3.7 Summary

PCV is the first scheme to exploit user mobility for bandwidth sharing applications in dynamic environments. PCV's architectural design can incorporate existing works on security, privacy, forwarding and routing. Extensive simulation results demonstrate the efficacy of our approach in terms of reducing bandwidth wastage. Moreover, through the results we address the issue of increasing bandwidth utilization, by introducing chunks and waiting times dynamically. In future, we plan to design human centric opportunistic applications deploying PCV. We believe that accurate prediction of contact volume paves the way for real-world deployment of interesting and useful applications, such as, content management and filtering, resource scheduling, bandwidth sharing, P2P gaming and video streaming in dynamic environments.

We presented the design and evaluation of a novel scheme called PCV that predicts contact volume to enable reliable and efficient data transfers in opportunistic networks. In particular, it focuses on two opportunistically meeting nodes in a participatory environment. In the next chapter we improve upon this idea and extend it to encompass entire source to destination paths.

CHAPTER 4

COSC: Paths with Combined Optimal Stability and Capacity in Opportunistic Networks

4.1 Introduction

In the previous chapter we presented the foundation work for efficient data transfers between two nodes. However, there are additional challenges faced when multiple nodes are considered simultaneously. For example, in participatory environments, it is common for source and destination nodes to be further apart, i.e., without a direct communication link between them. However, they may be connected via transient multi hop paths. Then it is natural to look for paths that are optimal in some sense. This chapter rigorously defines optimality and describes a scheme for identifying such paths.

A recent Cisco [67] report claimed 18 exabytes of mobile traffic in the cellular network in 2013, an increase of 81% compared to that in 2012. For years, cellular network providers have been looking into alternative solutions to off-load mobile data traffic. Femtocells, WiFi hotspots and opportunistic networking are among the popular options [68]. Although opportunistic networking is considered as a potential solution given the rapid adoption of mobile devices, there are a number of challenges related to its dynamic property in connectivity [69]. Typical *store-and-forward* solutions consider opportunistic networking as a way of best-effort content delivery, should an end-to-end path disappear. Several novel approaches proposed in the literature exploit *inter-contact times* and node mobilities to ensure packet delivery [7]. However, there is a need for further investigations to reduce delays and overheads in opportunistic networks (ONs).

We explore the idea of exploiting human mobility patterns to develop an effective algorithm for data forwarding in opportunistic networks. As users move around with their mobile devices, clusters of opportunistic networks are formed periodically among different social encounters. Users within these clusters can utilize the transient connectivity to distribute bandwidth intensive contents (prefetched video) among peers. Research works on human behavior and social connectivity have discovered that *humans follow a high degree of spatial and temporal regularity in their movements* [70] [71]. In this chapter, we propose a novel scheme that achieves combined optimal stability and capacity (COSC) in opportunistic networks. COSC exploits repeating trajectories embedded in the mobility histories of participating nodes. Whilechapterevious work [20] estimates contact volume that can be transferred during an *opportunistic contact*, the work presented in this chapter estimates capacity of an *opportunistic path.*

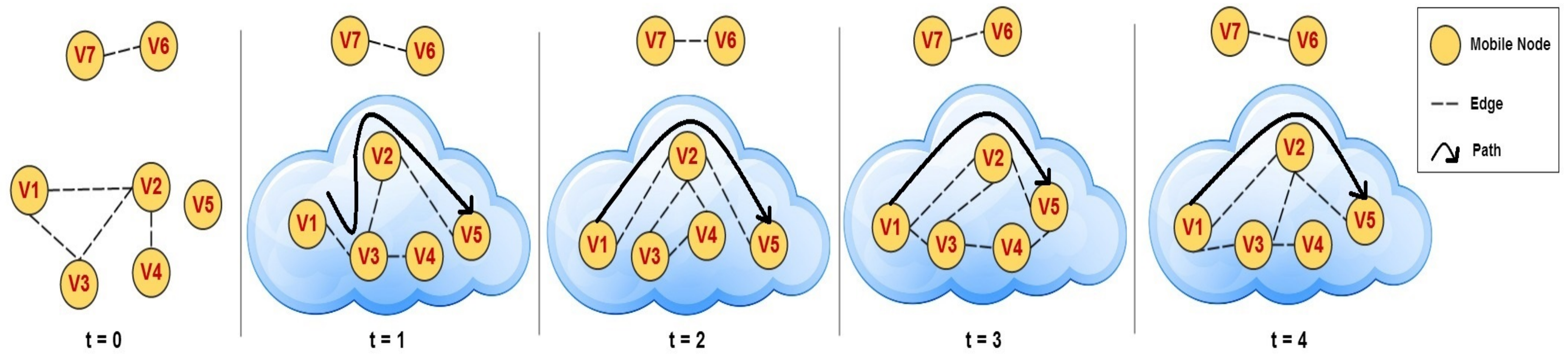


Figure 4.1: A TVG representation of an opportunistic network where one connected cluster $C_1, ...C_4$ comprising $\{v_1, ..., v_5\}$ is highlighted, with $a = 1$ and $b = 4$ and $a, b \in \tau[0, 4]$.. An example of a path set is also depicted, with a single changeover between $t = 1$ and $t = 2$.

A time varying graph (TVG) [72] is an instant snapshot of clusters of connected components in ONs. A TVG represents the connectivities between nodes over a series of discrete time intervals. In other words, TVG captures the pair-wise connectivity of any two nodes at different timestamps. If we relate this to the human mobility model, we can map devices' connectivities reoccurrence (as their users encounter each other) to a TVG over a period of time. At the start, mobile devices can retrieve the initial mobility graph, which is relevant to them, from a server. At run time, when users are moving around, their devices encounter each other. Information about such contacts at runtime can be used to verify and update initial mobility graphs. When a node needs to forward data, it searches for matching paths (containing its immediate neighbors) that lead to the desired destination. It is assumed that human mobility follows repeating trajectories, this predictive path lookup can reduce significant amount of overhead messages and result in packet delivery improvement. In COSC, each edge (or link) weight between two nodes of the TVG corresponds to the contact volume [20] when the two nodes come into contact. The effective capacity of a path from a source to a destination is the capacity of the bottleneck link. In addition to maximizing path capacity, it is also desirable to minimize the degradation in packet delivery due to path changes over a TVG. The utility of a path is a function of the path capacity and its stability. In effect, we solve an optimization problem that compares alternative paths in terms of their utilities over a series of snapshots of a TVG.

This chapter makes the following contributions:

1. Exploit mobility histories to identify suitable path for effective data transfer;
2. Propose solution to adapt to situations when a path does not exist in the mobility model;
3. Utilize capacity and stability metrics to evaluate available paths; and
4. Perform evaluation with synthetic and real-world mobility traces to demonstrate the feasibility of solution.

sectionSystem Design In this section, we discuss three aspects of our system design: (i) the graph theoretic model, (ii) the connected clusters within the TVGs, and (iii) the synthetic mobility models that allow more rigor performance evaluation.

4.1.1 Graph theoretic model

Dynamic environment of ONs can be viewed as edges appearing and disappearing in a time varying graph (TVG). Recently, Casteigts et al., have presented a unified model of time varying graphs [72]. We modify their general definition in this subsection to model ONs. Let the set of total nodes in the environment be V and let $E \subseteq V \times V$ represent the set of edges. Events, such as, inclusion and exclusion of edges happen over time $\tau \subseteq T$, where $T = \mathbb{N}$, i.e., discretized temporal domain. This work assumes the starting value of τ to be zero. ρ is the presence function such that $\rho : E \times \tau \rightarrow \{0, 1\}$. It represents presence of an edge at a given time - presence (absence) of an edge is represented by '1' (0). Therefore a TVG is represented as a tuple $G = (V, E, \tau, \rho)$. Figure 4.1 shows an example of a TVG, with seven nodes and τ in the range $[0, 4]$.

Edge computation in presence function: When two nodes v_x, $v_y \in V$ come within each others' transmission range D, they can form a communication link (edge in a TVG), i.e., when the predicate $\left\|pos(v_x) - pos(v_y)\right\| \leq D$ is true, where $pos(.)$ represents two dimensional position vector of a node. Therefore, the presence function ρ evaluates the aforementioned predicate at times $t \in \tau$ to determine if an edge is present between v_x and v_y in a TVG. The radio transmission link may be established using either Bluetooth or Wi-Fi Direct, which have their own characteristic D [53]. However, the experimental evaluation uses communication ranges of 100 and 150 meters, which are consistent with the ns-2 default values.

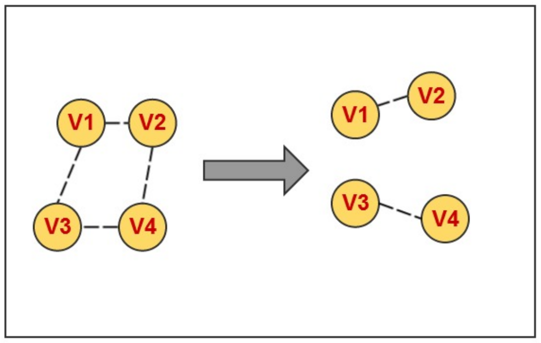
4.1.2 Time varying connected clusters

Zooming in on any one snapshot of a TVG ($t \in \tau$) reveals a single graph, that may or may not be connected. Though disconnectedness is not uncommon in snapshots derived from opportunistic networks, it is however easy to find connected clusters of nodes in it. Moreover, environments that have high node density are much more likely to contain such connected clusters, e.g., at a busy train station or a bus ride, wherein the nodes may be interested in sharing content such as a prefetched video.

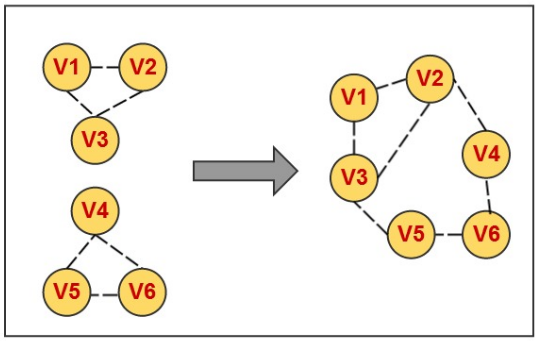
A cluster is essentially a subgraph in its parent TVG. A cluster retains its connectivity over a period of time, despite changes in the set of edges [73]. The lifetime of any given cluster is marked by starting and ending times $a, b \in \tau$. Consider one such cluster comprising a subset of mobile nodes $V' \subseteq V$. At any one starting point $a \in \tau$, there is an edge set E_a among these nodes. As the nodes move in time, the set of edges changes from E_a to E_{a+1}, then to E_{a+2} and so forth to an edge set E_b. The graph representing such a connected, time varying cluster comprising vertices V', but with varying edges is represented by $C_i(V', E_i)$ for $i = a, a+1, ..., b$. Figure 4.1 highlights one such cluster for $a = 1$ and $b = 4$.

The time stamp $b \in \tau$ and the edge set E_b marks the time after which a cluster comprising $V' \in V$ set of nodes fails to stay connected. Essentially, this happens whenever the connected set V' changes. The following three scenarios elucidate this event:

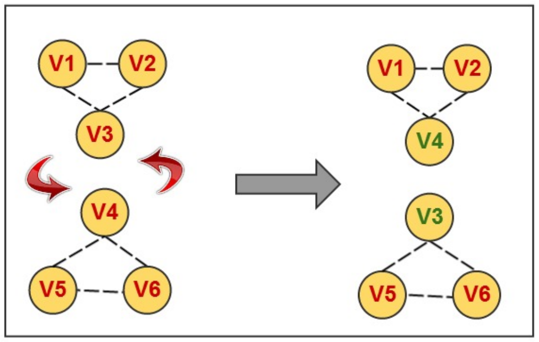
1. A cluster breaks down into smaller clusters, thus ending its lifetime and creating new ones as shown in Figure 4.2a.
2. Two or more clusters merge together to form a bigger cluster as shown in Figure 4.2b. This case ends the lifetime of multiple clusters to start a single new cluster.



(a) A cluster breaking into two; marks the end of one cluster and creation of two.



(b) Two clusters merging into one bigger cluster; ends two clusters and starts a single new cluster.



(c) The set of nodes changes, though the size is maintained; ends two clusters and starts two new clusters.

Figure 4.2: The three scenarios that mark the lifetime of a cluster

3. A cluster may lose and gain equal number of nodes till the next timestamp as shown in Figure 4.2c. Though such transformation preserves the number of nodes in any given cluster, it still changes the identity of the original set of connected nodes.

4.1.3 Synthetic Mobility Models

Human mobility patterns are key to understanding information flow in ONs. Fortunately, a number of human mobility traces have been collected in environments like, campuses, conferences, state fairs etc., that can be used in simulations pertaining to opportunistic networks [74] [75]. To bring more rigor to performance evaluation, synthetic mobility models are considered as well. Using traces generated with synthetic mobility models allow us to study the proposed solution in a variety of other scenarios.

There have been numerous efforts in determining the underlying common and stationary features of human mobility [12] [71]. It has been shown that the inter contact times in real-world mobility traces follow a power law distribution [12] as opposed to previously assumed exponential distribution. Moreover, it is also shown that various other statistical factors in human mobility, such as, flight lengths, pause times and fractal way-points follow power law distribution as well. Hence, to make the simulation results descriptive of real-world phenomena, we use self similar action walk (SLAW) as one of the synthetic mobility models [17] to generate movement scenarios. We have also used Home-cell Community-based Mobility Model (HCMM) as another synthetic mobility model [18] for trace generation. HCMM combines three main properties of human motion: 1) human movement is governed by social interactions; 2) users visit a few locations where majority of their time is spent; and 3) users prefer shorter paths over longer ones.

4.2 Path Selection

In this section, we deal with the design choices in order to find paths that enhance the information flow between a source and destination pair over a series of time stamps. Let P_i denote a multi hop (can be direct as well) path between two nodes of interest, in the time varying cluster C_i, for $i = a, a+1, a+2, ..., b-1, b$. The collection of these paths is also termed as a path set in this chapter. In order to find the most effective path set between a pair of connected nodes, we consider path capacity and path stability. Our objectives include:

1. Maximizing capacity between connected nodes; and
2. Maintaining a stable path.

COSC finds clusters in ONs and employs a utility function that considers path capacity and path stability to determine a path. A polynomial time, dynamic programming algorithm efficiently computes such paths in time varying connected clusters by maximizing the utility function.

4.2.1 Link capacities

When two mobile nodes v_x and v_y make contact for a duration of t^0_{xy}, the amount of data can be transferred depends on their velocity vectors [20]. Maximum amount of data that can be transferred between a pair of opportunistically meeting nodes v_x and v_y is given by $K(v_x, v_y)$.

In wireless communications, the received signal power S_{rec} at the receiver varies inversely with the distance d from the sender. The relationship is concretely represented as $S_{rec} \propto 1/d^{\psi}$, where the exponent ψ depends on the environment in which the nodes operate. It is also known that the transmission throughput depends on S_{rec}, making the throughput a function of instantaneous distance [50]. Our model [20] accounts for the changing distance and hence variable throughput during the contact period. It is interesting to note that an empirically obtained function of throughput against distance can be transformed into a time dependent function, by expressing the distance in terms of time for the moving nodes. If the time dependent throughput between v_i and v_j is denoted by $R(t)$ then the following equation couples it with the contact volume and contact duration:

$$K(v_x, v_y) = \int_0^{t^0_{ij}} R(t)\, dt. \tag{4.1}$$

The model in this chapter uses the contact volume $K(v_x, v_y)$ as the edge weight for the single hop connection between v_x and $v_y \in V'$.

4.2.1.1 Contact volumes on a path

Let e_k^i denote a single edge on a path P_i in an edge set E_i for cluster C_i. The two nodes forming the direct edge e_k^i, are represented by v_k, $v_{k+1} \in V'$. Then the edge weight is given by

$$w(e_k^i) = K^i(v_k, v_{k+1}). \tag{4.2}$$

The superscript i in the contact volume definition, represents value in the i^{th} time stamp. Hence the total effective capacity of a path $\xi(P_i)$ is the minimum of all the edge weights that make up that path,

$$\xi(P_i) = \min w(e_k^i), \quad \forall\, e_k^i \in P_i \tag{4.3}$$

As discussed at the start of this section, in order to improve the data transfers, it is desirable to look for paths that have a higher capacity. $\xi(P_i)$ is the precise mathematical quantity that our scheme tries to enhance.

4.2.2 Stability of paths

At the start of Section 4.2, we define P_i as a path in C_i for $i = a, a+1, a+2, ..., b-1, b$, then let $\psi(P_a, P_{a+1}, ..., P_{b-1}, P_b)$ represent the total number of changes, i.e., the points at which the identity of the path switches from the one in the previous instance of the subgraph. Formally, it is the number of indices i $(a \leq i \leq b-1)$ for which $P_i \neq P_{i+1}$. For example, considering the highlighted cluster in Figure 4.1, a chosen path $P_1 \in C_1$ from v_1 to v_5 is $(v_1 \rightarrow v_3 \rightarrow v_2 \rightarrow v_5)$, which is different from $P_2 = P_3 = P_4$ = $(v_1 \rightarrow v_2 \rightarrow v_5)$. Therefore, $\psi(P_1, P_2, P_3, P_4) = \psi(P_1, P_2, P_2, P_2) = 1$, as there is only one effective changeover in the path set.

In order to bring stability in the chosen path, it is desirable that it remains constant for as long as possible, despite the network dynamics. There are a number of reasons for such a desideratum:

1. *Overheads due to routing table updates:* A path switch requires exchange of messages to update routing tables, as nodes in a cluster, maintain routing information. This may also result in path oscillations [76].
2. *Data quality:* A switch to a new path entails, stalling the flow in the previous path and shifting the flow to the new path. This results in overheads, such as flushing buffers, and closing sockets, readers and writers, leading to poor data quality.
3. *Trust:* Protocols often need to establish trust whenever, new nodes are used for exchanging data, a costly procedure.

Therefore, in order to avoid the aforementioned penalties, a network designer should try to find paths that do not switch very often, i.e., minimize $\psi(.)$.

4.2.3 Finding a path set

Earlier, we identified two objectives for choosing paths in a connected cluster and modeled them quantitatively. We put them together using the following utility function,

$$utility(P_a, ..., P_b) = \sum_{i=a}^{b} \xi(P_i) - \theta \times \psi(P_a, ..., P_b) \tag{4.4}$$

$utility(P_a, ..., P_b)$ gives the total utility of the paths that are selected during the lifetime of a time varying cluster. Note that, the first term $\sum_{i=a}^{b} \xi(P_i)$ on the right side of Eq. 4.4 is the total sum of effective capacities all the paths, where each effective capacity of a path is the contact volume of the bottleneck edge making up the path. The second term gives the cost associated with path instability. The tuning constant θ can be used to weigh the second term according to network properties and application requirements. The unit of θ is MB and it is a quantity that signifies the amount of capacity that is equivalent to a single change over in the path set. A higher value of this quantity would suggest that every change over in the path set will incur a high penalty. The overall objective of this scheme is to maximize the utility given in by the Eq. 4.4.

It is possible to find connected components in linear time in a given snapshot of a time varying graph, i.e., at any $t \in \tau$ by using breadth first search or depth first search. Where the common technique of an outer loop can be used to cover all nodes in V.

In order to compute a path set of maximum utility for a time varying cluster, it is helpful to look at the available options for adding the final P_b. The following are the two options:

1. Choose P_{b-1} as the final path, which will add $\xi(P_{b-1})$ to the total utility, but will avoid the change penalty θ.
2. Choose a new path, that exists in C_b. It is natural to look for a path with maximum effective capacity, call it $P_{best} = P_b$. This path adds $\xi(P_b) - \theta$ to the final utility.

It is desirable to choose a path P_{b-1} in C_{b-1} that is also present in C_b, i.e., $P_b = P_{b-1}$. This will avoid the change penalty θ. The effect of C_b on the earlier part of the solution can be anticipated using dynamic programming.

Let $Opt(i)$ denote the solution to the subproblems for the clusters $C_a, ..., C_i$. To compute $Opt(b)$, i.e., a solution for the complete time varying connected cluster at hand, one should look for the last changeover that occurred in the path. Let the last changeover be between C_i and C_{i+1}. This means we have a path P_{i+1} in all the clusters $C_i, ..., C_b$. Therefore, the edges of P_{i+1} are present in all of those clusters. Let $common(C_i, ...C_j)$ represent a graph that is an intersection of all the clusters $C_i, ...C_j$ and let the best path in such a common graph be $P_{best}(i, j)$ for $a \leq i \leq j \leq b$. Then, if the last changeover occurred between the indices i and $i + 1$, the recurrence relation for the optimal utility can be expressed as $Opt(b) = Opt(i) + (b - i) \times \xi(P_{best}(i + 1, b)) - \theta$. However, there is a special case that may exist for finding a single path for the entire duration of a cluster that incurs no changes i.e., $\psi(.) = 0$. In this case $Opt(b) = (b - a + 1) \times \xi(P_{best}(a, b))$. Hence the final recurrence that gives maximum utility, set of paths and guarantees the minimum number of packets transferred in a cluster is as follows,

$$\begin{aligned} Opt(b) = &\max\{(b - a + 1) \times \xi(P_{best}(a, b)), \\ &\max_{a \leq i \leq b}(Opt(i) + (b - i) \times \xi(P_{best}(i + 1, b)) - \theta)\} \end{aligned}$$

The algorithm presented above first computes for each pair i, j, the $common(C_i, ...C_j)$ and $P_{best}(i, j)$ values for $a \leq i \leq j \leq b$. There are $O((b - a)^2)$ such pairs and the complexity to compute each subgraph is $O(|V|^2 (b-a)^2)$, where $|V|$ is the number of nodes in a cluster. The factor of $|V|^2$ arises because of the maximum number of possible edges in a cluster. A simple linear search is employed to compute the best path in each graph in linear time. Therefore the total running time of the algorithm is $O(|V|^2 (b - a)^3)$. The algorithm can be speeded up by computing the graphs $common(C_i, ...C_j)$ and $P_{best}(i, j)$ for a fixed value of i in order of $j = i, ..., b$. Then the total polynomial running time of the proposed algorithm is reduced to $O(|V|^2 (b - a)^2)$.

4.3 System Architecture and Implementation

In order to implement a software agent that may execute our scheme and compute a path set of maximum utility, we present a system architecture developed for Android devices. Figure 4.3 depicts the proposed modular design consisting of two major components.

We make use of Android system services to get instances of Wi-Fi P2P manager and other sensor controllers. The Wi-Fi P2P manager is responsible for discovering and setting up connection to nearby neighboring devices to exchange control information, contact volume estimates and data streams. Sensors such as accelerometer, gyroscope and GPS

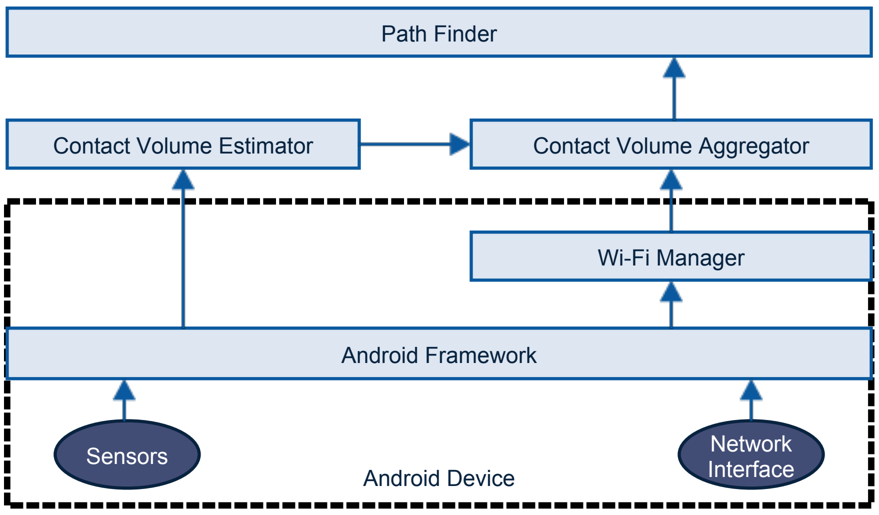


Figure 4.3: System Architecture

feed information to the contact volume estimator. All of the sensor data is made available through system calls executed through the Android framework.

4.3.1 Contact volume estimator and aggregator

The contact volume estimator receives sensor data such as device's accelerometer readings, along with GPS information to estimate the contact volume between neighboring devices. The basic idea hinges on the fact that, two mobile devices have a high contact volume if the contact duration is long. The estimator predicts the contact volume based on the data rate profile as well as the device's estimated contact duration inferred from its velocity and initial position vectors.

The device's own contact volume estimation is compared against the one received from neighboring devices and thus aggregated using a pessimistic approach, i.e, the minimum of the two contact volumes is chosen as a final estimate for further computation. It is possible for the two devices to predict slightly different contact volumes, as it is not a symmetric measure [20]. Formally a device v_x computes contact volumes $K(v_x, v_i) \;\; \forall v_i \neq v_x$,

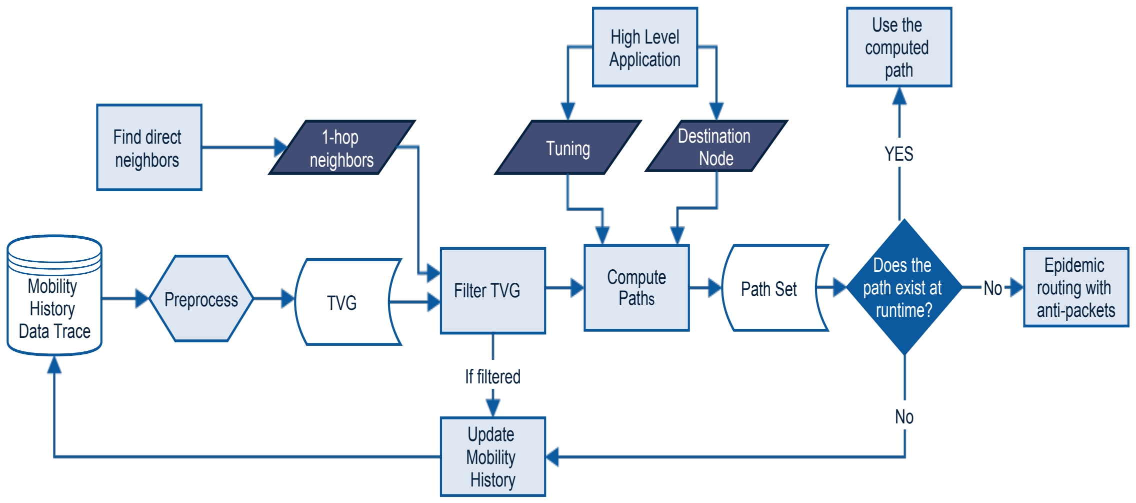


Figure 4.4: Path finder's flowchart. The tuning parameter is provided by a high level application. The path finder adaptively switches between using maximum utility path set and epidemic routing with anti-packets.

where $v_i \in V'$ is an immediate neighbor of v_x. Similarly, the neighboring devices estimate their contact volumes for device v_x, i.e., $K(v_i, v_x)$ and send this control information to v_x. Subsequently v_x chooses $\min(K(v_x, v_i), K(v_i, v_x))$ for all neighboring v_i. Also, note that the neighboring devices of v_x are responsible for sending contact volume information pertaining nodes that are not in direct connection with v_x.

4.3.2 Path finder

The path finder module is responsible for carrying out all the necessary tasks to compute a maximum utility path set based on the mobility history of the users comprising the opportunistic network. Figure 4.4 shows a flow chart that describes all the important processes carried out by the path finder module. Initially the device downloads a relevant mobility history data trace and preprocesses it. The preprocessing step ensures cleansing of the data and turns it into a usable form. For example, mobility history data traces can be in the form of waypoints, location coordinates or mere device to device contacts. As all of these forms of data have associated time stamps, preprocessing helps in building a TVG.

The path finder periodically acquires information about its direct one hop neighbors through a simple device discovery process. Knowledge of direct one hop neighbors at runtime helps in partially verifying the TVG and saves unnecessary computation.

The node can verify whether its one hop neighbors depicted in the TVG are also its one hop neighbors at runtime, thereby preventing computation of false paths early on in the path finding process. If the one hop neighbors do not match with the ones in the TVG, those nodes can be removed along with their edges before the path computation. Moreover, the history is updated to reflect a more recent view of the network.

A high level application, that is not part of the path finder module provides the destination node and tuning parameter θ to appropriately penalize the change overs in a path set. The path finder then computes a path set based on the processed TVG, to maximize the utility for the given destination node. Though repetitive human movement patterns favor this approach and guarantee a high hit rate, there is still a chance, that the computed paths based on mobility history may be nonexistent at runtime. As humans sometimes do deviate from their usual movement patterns, COSC verifies once more whether the computed path set is pertinent. If it is, then the computed path set is deemed fit and subsequently used for data transfers. On the contrary, our scheme adapts and falls back to epidemic routing with anti-packets [77], wherein, the source *infects* its immediate neighbors with the data packets; the neighbors then send the data packets to their next hop neighbors. This process of data packet infection continues till the destination receives the data, at which point, it sends back an anti-packet as an acknowledgment. All the nodes that are earlier infected, use the anti-packets to purge the corresponding original data packets from their buffers. Finally, COSC has a corrective mechanism and in the case when epidemic routing is invoked, it learns the new movement patterns and updates the original movement history.

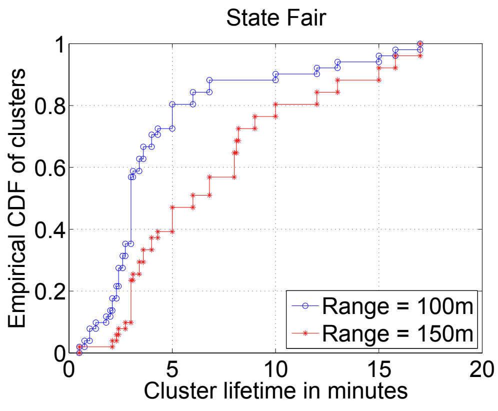
4.4 Evaluation

This section evaluates the performance of COSC. First, the distribution of connected time varying clusters is presented. We then evaluate the total information flow that is possible in opportunistic networks. Finally, based on information flow results, we compare COSC with shortest multi hop paths and investigate the improvement achieved by COSC for file transfer failures.

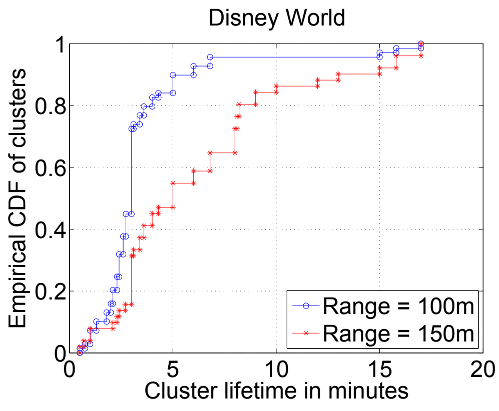
4.4.1 Simulation setup

For the purpose of this simulation, both real-world and synthetic mobility traces are considered. The real-world mobility traces were contributed by two groups of volunteers who visited a State Fair and Disney Land in Orlando respectively. The positions of the individuals were logged at 30 second intervals for a duration of approximately three and twelve hours. Therefore the positions of all the users at each time instance (every 30 seconds) is referred as a snapshot. There are a total 19 and 41 log files for State Fair and Disney Land traces respectively, where the i^{th} log file represents the position of that user in two dimensional space for the entire duration of the collected trace. In order to simulate a user-based opportunistic network, we assume, that all log files represent users carrying mobile devices that are capable of making connections with each other if they come within the radio communication link.

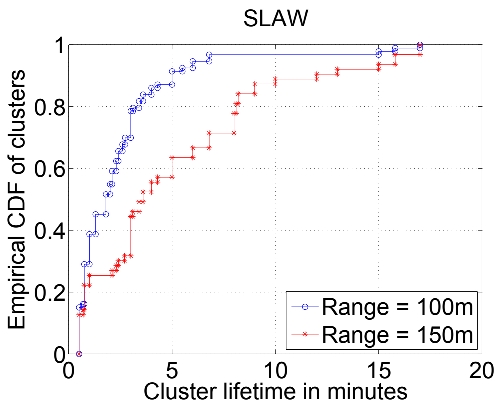
Apart from the real-world mobility traces, we use Self-Similar Least Action Walk (SLAW) mobility model for generating additional scenarios that captures the statistical human mobility characteristics. We span the simulation area to a $500m \times 500m$ two dimensional space, and make use of both slow and fast moving nodes (10 each). The details of this synthetic mobility trace are provided in Table 5.2. Furthermore, we have also used the HCMM mobility model to test our scheme with 20 nodes [18]. The details of parameters used to setup the HCMM trace are presented in Table 4.2   We have used 100 m and 150



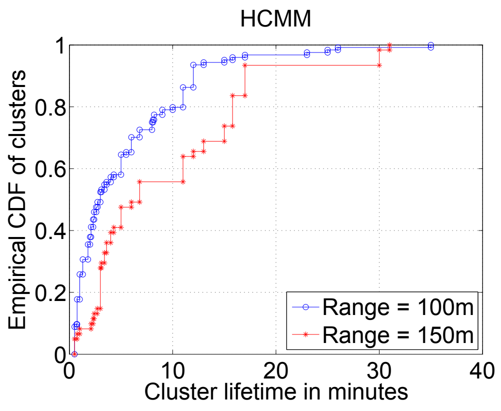
(a) State Fair - Longer communication range produces clusters with longer lifetimes.



(b) Disney Land



(c) SLAW - Short lived clusters are most prevalent



(d) HCMM - Clusters living for over 30 mins found as well

Figure 4.5: Empirical CDF of the lifetime of clusters in different models

m as the two radio communication ranges. The specific values are used to depict ranges of Wi-Fi Direct [53], which is a prevalent technology in the smart phones and mobile devices of today.

4.4.2 Connectivity of time varying clusters

In order to find suitable path sets for enhanced information flow in a number of applications, there is a strong need to find connected components in several human mobility scenarios. With the aforementioned setup in Section 4.4.1, we ran depth first search on each of the snapshots of the time varying graphs to first find the individual connected com-

Table 4.1: SLAW mobility trace used for simulation

Description	**Slow**	**Fast**
Exponent of step length distribution	1.6	
Exponent of pause time distribution	0.6	
Hurst Parameter	0.75	
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 4.2: HCMM mobility trace used for simulation

Parameter	**Value**
Velocity of node	min: 1m/s, max: 5m/s
Nodes	20
Groups	5
Rewiring probability	0.2
Travelers	5
Grid	4 × 4
Simulation area	500 × 500m

ponents in linear time. Then in order to find clusters that adhere to the definition provided in this chapter, i.e., follow the lifetime rules of a cluster, each of them was backtracked till the identity of the nodes did not change. As the traces have been generated with 30 $secs$ time intervals, we did not include clusters, that just remained connected for a single snapshot of a time varying graph. Figure 4.5 shows the empirical CDF plots obtained for State Fair, Disney World, SLAW and HCMM mobility traces. The plots, depict the CDF of the life time of various clusters. It is observed that, for a communication range of $100m$, the real-world traces contained most of the clusters that remained alive for approximately $3\ minutes$. As approximately 65% of the clusters disintegrated after this time interval. However, the SLAW mobility trace shows a much more transient behavior, where approximately 80% of the clusters remained connected for less than $3\ minutes$. We believe it is because of the inclusion of 10 fast moving nodes, which are highly dynamic and thus, disrupt the connectivity among other connected nodes. Note that, the definition of time varying connected cluster presented in this work is strict, as even an inclusion of a single node to an already connected cluster, starts the lifetime of a new cluster and ends that of an earlier one. Data trace based on HCMM showed some clusters that were alive for up to 30 minutes. This is probably true because of the cell based communities present in the trace. For a communication range of $150\ m$, the clusters are observed to remain working for longer time intervals. However, in the SLAW mobility trace, we do not see the effect as pronounced as in the real-world or HCMM traces as 60% of the clusters remained alive for less than $5\ minutes$.

4.4.3 Possible data transfer

To quantify the possible data transfers between nodes in a cluster, we randomly select a pair of nodes whenever a cluster is formed. Based on COSC, we select a path set for that particular pair and evaluate the contact volume of each path. The contact volume of a single path is chosen to be the one that makes up the weakest link between the two end nodes. Its value is valid for a single snapshot of the cluster at hand. Finally, the total contact volume for the entire path set is computed by summing individual minimum contact volumes. For example, back in Figure 4.1 we found a path set $\{P_1, P_2, P_3, P_4\}$. Suppose the contact volumes of the weakest links in each of these paths are $10, 20, 20$ and $10\ MB$ respectively, where each path is valid for $30\ secs$ interval, then the total contact volume will be the sum of these quantities, i.e. $60\ MB$. Note that, though the paths P_2, P_3, P_4 essentially represent the same path, the contact volume of the weakest link maybe different, even when the contact durations are for $30\ secs$. The reason for that was investigated in our previous work [20], where it is shown that the contact volume not only depends on contact durations, but also on instantaneous distance among communicating nodes. The simulation is repeated 1000 times, and the averages are plotted in Figure 4.6, for three mobility traces. The error bars represent the standard deviation. It is observed that an average of approximately $125\ MB$ can flow at best in a State Fair setting with $150\ m$ radio link. However, SLAW shows the minimum average contact volumes on a path set, with $45\ MB$ at a $100\ m$ range. HCMM showed the least percentage change in the contact volume of a path, when the radio range is increased from $100\ m$ to $150\ m$.

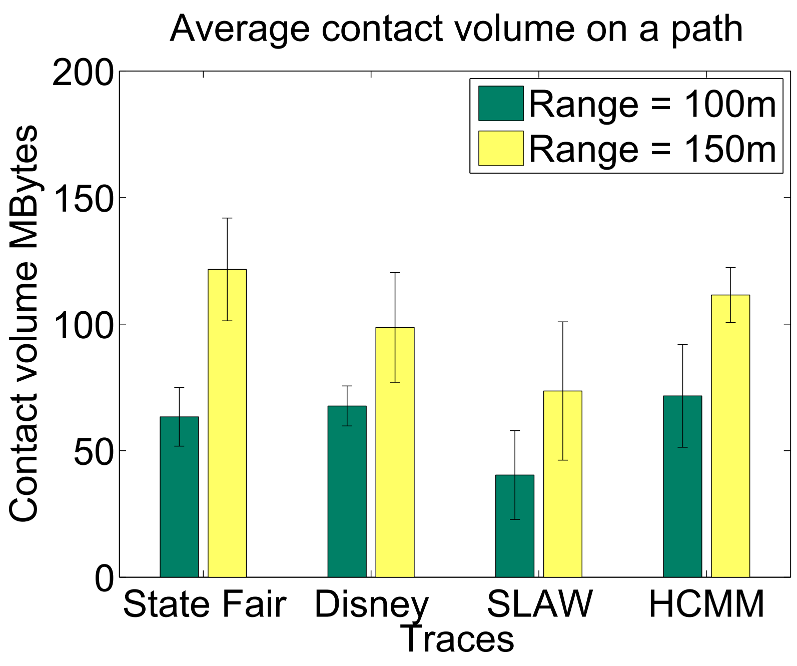
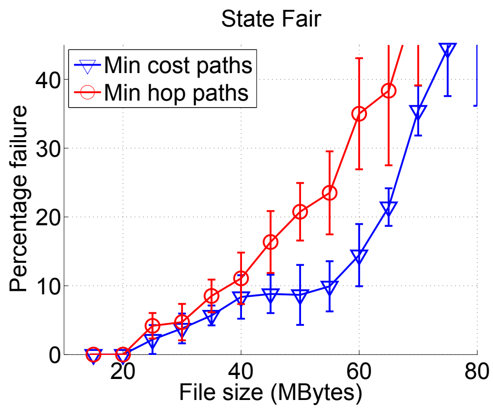


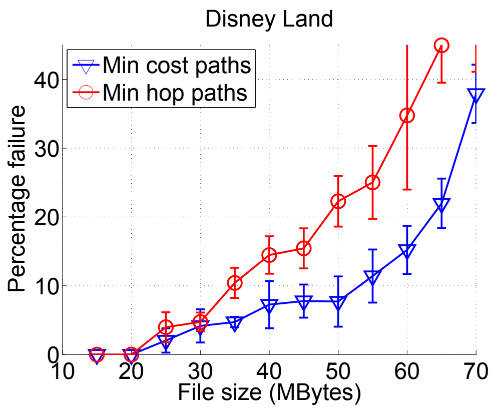
Figure 4.6: Average contact volume on a path set between arbitrarily chosen nodes in clusters. Longer communication range opens up the opportunity for transferring more data.

4.4.4 File transfer failure rates

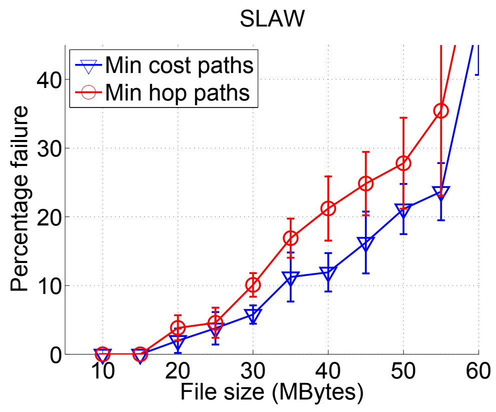
In order to compare the maximum utility path set, obtained using the methodology described in this chapter, against the well known shortest multi hop path set, we use file transfer failure rates as a measure. The shortest multi hop path set is formally defined as a path set in a time varying connected cluster, where all link costs are assumed to be positive



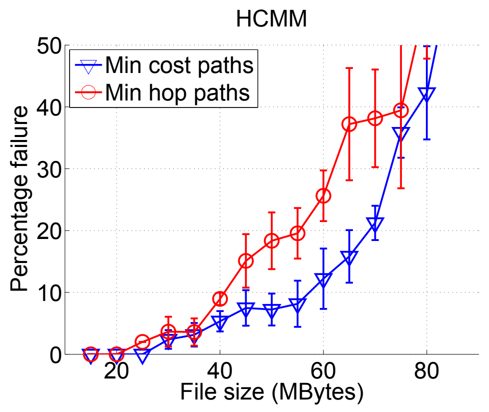
(a) State Fair - The difference in the file transfer failure rates are most pronounced around file sizes close to the contact volume, i.e., 55 MB



(b) Disney Land - The maximum difference in failure rates is seen around 60 MB files



(c) SLAW - Failure rates climb up fairly quickly after the possible contact volumes. Failure rates of min cost path are 40% less at best.



(d) HCMM - Failure rate is lesser compared to other traces at 70 MB file size.

Figure 4.7: Percentage of file transfer failure rates at communication range of $100\ m$. A comparison of COSC against shortest paths computed based on number of hops

and equal to 1. Thus a shortest multi hop path set, contains the shortest paths in terms of hop count for each snapshot of a cluster. For the purposes of this study, we ran the simulation a 1000 times and the average file transfer failure rates were plotted against file sizes. The error bars represent the standard deviation. Failure rate is defined as the percent of file transfer failures. In other words, failure rate is given by number of *(failed transfers) / (total number of attempted transfers)*. The range of file sizes is chosen based on results shown in Figure 4.6, i.e., file sizes were picked in the proximity of the maximum possible amount of data transferrable for each of the three traces at 100 m range. The rationale for doing so is explained below. For example, if huge file sizes are used in comparison to the found data limits, it would invariably lead to failed file transfers. On the other hand, using file sizes that are very small, will let them through almost all the time. Therefore, both the extremes do not help in revealing any interesting information about our methodology. Figure 4.7 depicts the comparison of failure rates between COSC against the scheme that uses shortest multi hop path. Though we compared the two schemes for both 150 m and 100 m communication ranges, the plots are drawn only for 100 m, because higher ranges reveal similar trends, but around larger file sizes. For real-world mobility traces, it is observed that the file transfer failure rate of COSC is approximately 60% *less* than that of shortest multi hop for file sizes around 55 MB. Whereas, the advantage in SLAW mobility trace is almost 40% at best. Moreover, HCMM shows the least failure rate for 70 MB file sizes, where the other three environments have a 30% or more failure rate.

4.5 Summary

This work presents a novel methodology for modeling information flow in user-based opportunistic networks, that are common to pervasive environments. The network is initially viewed as a time varying graph, with a presence function that defines transient edges based on pairwise contacts. We then present a complete and thorough definition of time varying connected clusters and define scenarios for their lifetimes. The simulation results pertaining to connectivity in clusters validate this intuition, as humans are social beings and are often observed to interact in groups. To the best of our knowledge, this is the first scheme to estimate contact volumes over paths in participatory environments.

Keeping in mind the dynamic nature of participatory environments and time varying clusters, we identify two opposing cost objectives that aim to minimize the number of packets transferred and induce stability in the information flow paths between nodes. We then present a polynomial time algorithm to solve the resultant cost function. The simulation results show that using our methodology, nodes can transfer from $45\ MB$ to $120\ MB$ of information in an opportunistic environment. Furthermore, a reduction of up to 60% in file transfer failure rate is seen in comparison to shortest multi hop path scheme.

The work described in this chapter makes efficient use of capacity and stability over paths for transferring data effectively in participatory environments. The capacities over a path are modeled using PCV for pairwise contacts between participating nodes. In addition to the capacities, this work incorporates the notion of stability to avoid jitter and frequent path switching for improved user experience at the application layer. In the next chapter, we will digress a little, and look into mobility of nodes to improve communication and collaboration among participants.

CHAPTER 5

MOEME: Real-Time Mobility and Orientation Estimation for Mobile Environments

5.1 Introduction

In the previous chapters, we presented schemes for effective data transfers among nodes in participatory environments. While discussing COSC, we pointed out that humans follow spatial and temporal regularity in their movements [70] [71]. This lead us to explore human mobility further and find ways to use enhance data transfers. In this chapter, we present our findings and subsequently develop a lightweight and distributed mobility estimation scheme that aids our goal of making communication and collaboration more efficient.

As mobile environments rely primarily on user mobility as a mechanism for transporting content and data in general, identifying and modeling user mobility provides researchers and network designers with key insights for improving performance, efficiency and productivity. For example, epidemic routing [13] guarantees shortest latencies in message delivery, however, knowledge of user mobility and diffusion has been exploited by various schemes to considerably cut down energy expended in already resource constrained devices with a small tradeoff on delivery times [14] [15] [16]. With the potential of similar conceivable advantages there are several works which focus on mathematically modeling the human mobility patterns [78]. In another work, researchers characterize pause times, inter-contact times and speeds of users based on real world traces collected over different periods of time [79]. Recently there has been an effort to model user mobility where the authors use small areas, in which the user mobility does not affect communication, as building blocks for a more complex queuing model [80]. However, existing schemes that estimate mobility rely on repetitive user patterns and user trace data. Information about trajectories and times of user movement are critical. Moreover, if users deviate from regular patterns of their movement or if no such patterns exist, which is very common in open environments such as parks, malls or streets, the models start falling apart, thereby leading to a serious limitation in guaranteeing accuracy.

To this end, we present a novel scheme called MOEME: Real-Time Mobility and Orientation Estimation for Mobile Environments. To the best of our knowledge, this is a unique contribution to estimate mobility of users in mobile environments in real-time without i) a priori knowledge of user movement patterns, ii) a Global Positioning System and iii) infrastructure support. The key insight which allows MOEME to perform well lies in its message exchange mechanism. MOEME employs the concept of temporal distances - devices exchange information about their temporal distance to other devices. We show that gathering this information for users in real-time is easy and incurs small space complexity. Coupling this information with the model learned using logistic regression, MOEME powered devices are able to make estimations about user mobility in real-time. Furthermore, we see that model learned for MOEME generalizes well and is tested to make accurate estimations across both real-world and synthetic traces in the presence of haphazard and random user movements.

MOEME empowers both users and system architects with the knowledge of user mobility. MOEME estimates relative directional mobilities of all the users in a mobile environment, in addition to counting the number of users present within a desired spatiotemporal radius. MOEME can also be employed to predict the absolute orientation of the users in a mobile environment. We demonstrate how MOEME can be used in a variety of scheduling and resource allocation applications. For example, in our recent work we predict the 'contact volume' i.e., the maximum amount of transferrable data between two opportunistically meeting mobile nodes [20]. The accuracy of predicting contact volume can be improved directly with the knowledge of directional mobility of users. Furthermore, it helps in predicting the total volume of information that can be pushed from one end of the network to the other or from a particular source node to a desired destination node.

The novelty of MOEME lies in its ability to estimate user mobilities with no *a priori* knowledge of movement histories. MOEME can be employed in indoor as well as outdoor environments. MOEME is fully distributed, light weight and has a time complexity of $O(n)$ at each node, where n is the number of nodes present in the mobile environment. Moreover, MOEME does not rely on Global Positioning System (GPS) or other location tracking systems [81] [82], which may be power hungry and thus limit the practicality of the technique.

In our previous work [83], we devised a distributed scheme to estimate the relative directional mobilities and the number of users in a desired spatiotemporal region. Whereas, in this chapter we have refined the scheme significantly to estimate the absolute orientation of users in mobile environments.

5.1.1 Contributions

Major contributions of MOEME include:

1. Estimation of relative directional mobilities of all users in real-time without requiring their movement histories.
2. Estimation of the number of users, that are likely to be within a spatiotemporal region of interest.
3. Predict the absolute orientation of the users in a mobile environment.

To the best of our knowledge MOEME is the first scheme to make the above mentioned contributions for estimations of user mobilities in mobile environments.

MOEME estimates distance of users and number of users within a distance of 300m with an accuracy of 89%. The estimates of direction of users are 77% accurate for nodes within 200m.

5.2 Related Work

In dynamic and pervasive networks, a significant amount of research has focused on efficient routing schemes [37] [84] [85] [86] [87] and content dissemination frameworks [88] [89] that exploit repetitive patterns in human movement. Recently, there has been some focus on content and service distributions in open environments, such as parks, malls etc. where history from past visits is not available [14] [44] or repetitive patterns do not exist at the time scale of few minutes (0 to 15 minutes). There is only one work [47] that aims to create plausible mobility merely through user contacts but requires a centralized server to keep track of all contacts in real time. In contrast, we present the first work that estimates 1) relative directional mobilities of all users in real-time without requiring their movement histories and 2) the number of users, that are likely to be within a spatiotemporal region of interest, through opportunistic contacts in a completely decentralized manner.

5.3 System Description

5.3.1 Types of nodes

We represent the set of all nodes in the pervasive environment by $V = \{v_1, v_2, v_3, ..., v_n\}$. Among these n, there are k nodes of interest (NOI). We represent the set of NOIs by P, where $P \subseteq V$. These nodes of interests are used to make predictions about the direction of motion for the rest of the nodes in the system. The nodes of interests can be static or mobile. Each $v_i \in P$ makes a prediction for the nodes in the set $V - \{v_i\}$, the set of all nodes except v_i.

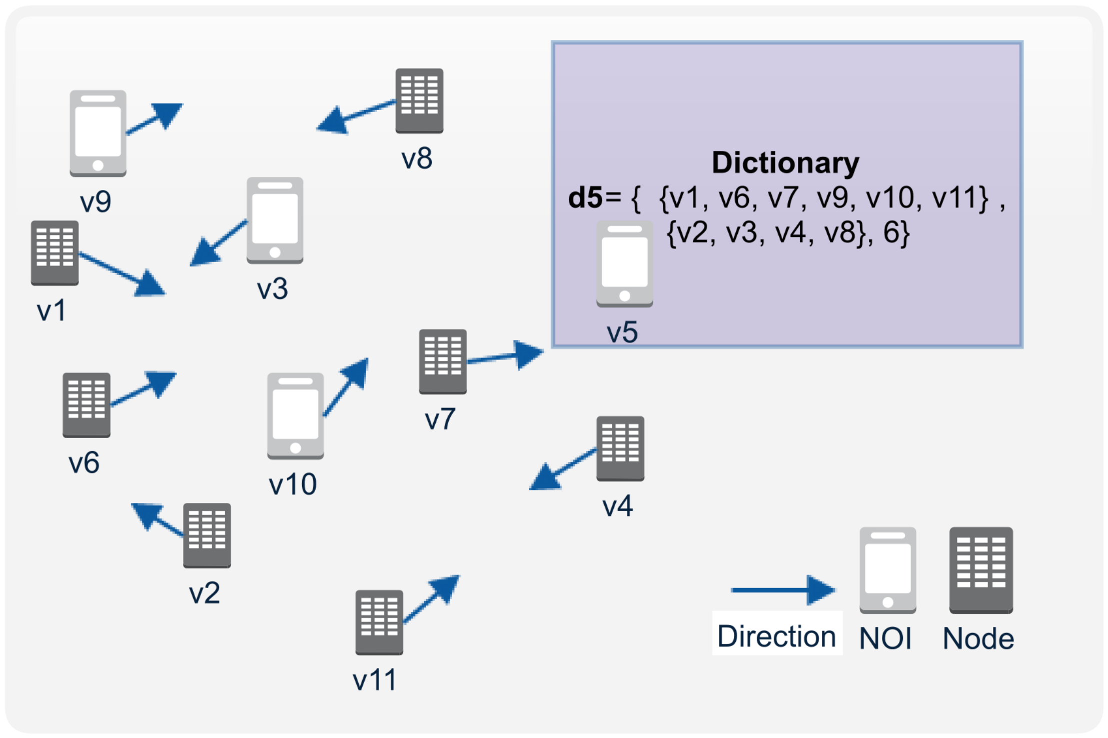


Figure 5.1: A mobile environment

5.3.2 Dictionaries at NOIs

Every node $v_i \in P$, maintains a dictionary d_i to keep track of the following information in real-time:

1. Set of nodes moving closer: C,
2. Set of nodes moving away: A, and
3. Number of nodes moving closer: μ.

The structure of the dictionary is $d_i = \{C, A, \mu\}$, where $C \subseteq V$, $A \subseteq V$ and $C \cup A = V - \{v_i\}$

The Figure 5.1 depicts a scenario with several regular and monitor nodes. It also shows the contents of dictionary maintained by $v_5 \in P$.

5.3.3 Underlying network

The underlying network on which MOEME is built on is inherently opportunistic [68]. The network may comprise of a mix of both static and mobile nodes. However, only the nodes that are within each other's communication range can exchange messages. The purpose of this work is not routing or message forwarding to a destination node. MOEME uses the information it acquires from other nodes, for local and real-time computation of directional mobilities of other nodes in the environment.

5.3.4 Temporal distance

To keep track of an approximate measure of distance among nodes in a network, the notion of temporal distance is used [14]. It also gives a measure of how fast information can travel among nodes by means of transitive connections among them. At each time instant t, every node in the network maintains a *timer* (Section 5.3.5) value for the rest of the nodes in the network. The timer values give the measure of temporal distance. However, timer values are not symmetric between nodes due to their distributed nature. It is possible that at some time instance t_0, the timer value that node v_i records for v_j is different from the one v_j holds for v_i.

5.3.5 Timer update

Let $t_{v_i}(v_k)$ denote the time elapsed since node v_i made contact with node v_k, where $t_{v_i}(v_i) = 0$. Local timer values for each node are incremented after every time unit, (e.g., 30s, 60s etc. depending on devices). When two nodes v_i, v_j come into *contact* (within transmission range), v_j updates its timer values according to the following rule: $\forall v_k \neq v_j$: $t_{v_i}(v_k) < t_{v_j}(v_k) - t_{av}$, set,

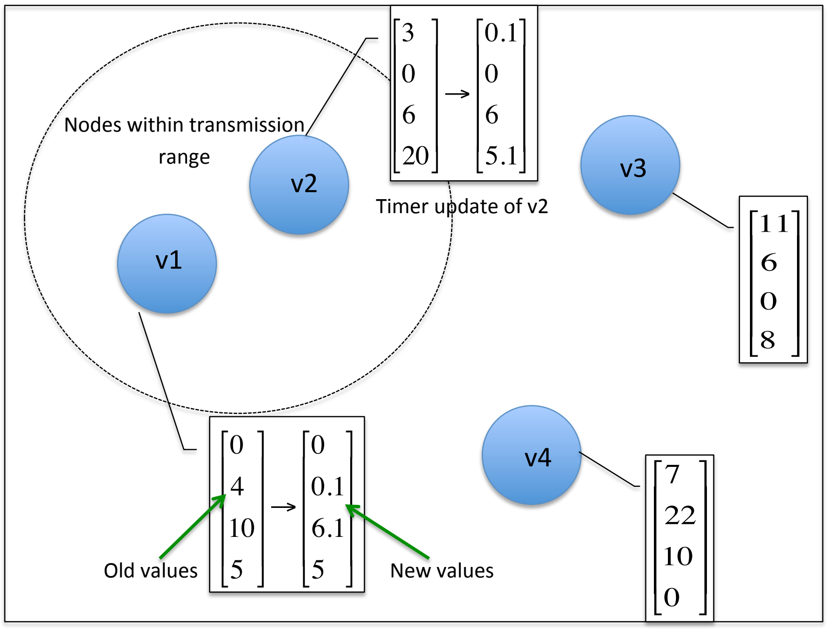
$$t_{v_j}(v_k) = t_{v_i}(v_k) + t_{av} \tag{5.1}$$

$$\tag{5.2}$$

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. 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 is a different definition of t_{av} from that in [30] where the value of average time required to travel between the two nodes is included. A small value of t_{av} is used as the physical distance between the connected nodes is of little importance. t_{av} captures the concept of nodes being in contact through other nodes in between. Therefore, value of $t_{av} > 0$ creates a gradient in a connected subset of nodes. Figure 5.2 shows an example of how the timer values are updated according to rule 5.2, when two nodes come within each others' transmission range. In this case, the network comprises of four nodes. Each node maintains timer values for itself $t_{v_i}(v_i) = 0$, and the nodes in the rest of the network. t_{av}, is set to be equal to 0.1 for the purpose of illustration. The timer values at each node v_i, where $i = 1, 2, 3$ and 4 are shown in a vector of the form $\begin{bmatrix} t_{v_i}(v_1) \\ t_{v_i}(v_2) \\ t_{v_i}(v_3) \\ t_{v_i}(v_4) \end{bmatrix}$.

5.3.6 Additional timers at the time of contact

MOEME also employs two additional arrays of timers **MT** and **RT**, at each node $v_i \in V$. Both the arrays contain information that depicts the view of the network each v_i observes, just before the time of contact with another node $v_j \in V - v_i$.



Nodes within transmission range
v1
v2
v3
v4
Timer update of v2
$\begin{bmatrix}3\\0\\6\\20\end{bmatrix} \rightarrow \begin{bmatrix}0.1\\0\\6\\5.1\end{bmatrix}$
$\begin{bmatrix}11\\6\\0\\8\end{bmatrix}$
$\begin{bmatrix}0\\4\\10\\5\end{bmatrix} \rightarrow \begin{bmatrix}0\\0.1\\6.1\\5\end{bmatrix}$
$\begin{bmatrix}7\\22\\10\\0\end{bmatrix}$
Old values
New values

Figure 5.2: Timer updates

1. **MT** records the values of all the timers that node v_i has for all the other nodes in the network, just before the time of contact.
2. **RT** contains the values of timers that some other node v_j brings along with it for all the other nodes in the network, just before contacting v_i.

At the time of contact, first the values in **RT** and **MT** are updated and thereafter, the values for regular timers defined in section 5.3.5 are refreshed according to the update rule 5.2. Both these arrays support the usual indexing operation, where the index represents the identity of a particular node, e.g. **MT[3]** shall access the value in **MT** for node number 3.

5.3.7 Time since last contact and threshold parameter matrix

Each of the nodes $v_i \in V$ record the time elapsed t_l, since it last contacted some other node in the network. When a node v_i comes into contact with another node v_j, t_l is set to zero for both the respective nodes. Hereafter, the value of t_l keeps climbing with incremental time units until another contact occurs for a node.

Each of the nodes also has a copy of threshold parameter matrix Θ, which is unique to a type of network and is obtained through the machine learning technique, logistic regression. The rows of Θ are the individual threshold vectors corresponding to each time since last contact, i.e. integers values of $t_l \in [0, 10]$. The row values of Θ, timer arrays **MT** and **RT**, and time since last contact t_l are together used by $v_i \in P$ to make a real-time prediction about the relative directional motion of the rest of the nodes in the network.

5.4 Learning the threshold parameter matrix

In MOEME, each node $v_i \in P$ strives to estimate the relative direction of motion of the other nodes in the network. In order to do this successfully, these NOIs keep a copy of the threshold parameter matrix. MOEME learns this parameter matrix Θ offline by applying logistic regression on data extracted from mobility traces. The novel aspect of the proposed scheme is the fact that it is completely distributed in nature. NOIs can locally make estimates for the directional mobility of other nodes in the network. There is no centralized agent, thereby making the scheme robust and fault tolerant.

5.4.1 Logistic regression

The objective of machine learning algorithm logistic regression, is to produce a threshold parameter vector θ that minimizes an appropriate cost function $J(\theta)$ for a subset of the given data [90]. The algorithm was run a number of times, each time on a different subset of the original data. The original data is filtered according to the t_l, the time since last contact values, to obtain subsets of the data, where t_l takes integers values in the range [0, 10]. All the obtained θ vectors are transposed and stacked together to form the threshold parameter matrix, Θ. At the time of directional mobility estimation MOEME selects an appropriate row from the matrix Θ based on t_l and processes it with the feature vector x, where the feature vector is calculated using both **MT** and **RT** . Logistic regression serves us well because the problem at hand falls under binary classification problems [91].

5.4.1.1 Sigmoid function

It has been shown that a naive linear hypothesis

$$h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1 \tag{5.3}$$

where,

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \; and \; x = \begin{bmatrix} 1 \\ x_1 \end{bmatrix} \tag{5.4}$$

, will work poorly in the case of classification problems, although it works well for linear regression [91]. Therefore, in order to get a better estimate, the sigmoid function is used and the hypothesis is defined as

$$h_\theta(x) = g(\theta^T x) \tag{5.5}$$

where,

$$g(z) = \frac{1}{1 + \exp^{-z}} \tag{5.6}$$

5.4.1.2 Cost function and gradient

The data set used comprises two columns, x_1 and y. At each time instance, for each pair of nodes, x_1 and the corresponding y values are calculated. y can take values either

0 or 1, where 0 means the nodes in consideration are in reality moving away, whereas 1 means they are moving closer. We use the notation x^i and y^i to denote the i^{th} example in the training data set, i.e. the i^{th} row in the data set. Simply minimizing the sum of the squares of errors, where errors are usually defined as a difference in the actual y and the guessed values, shall not work [91]. Logistic regression algorithm fails to find a suitable global minima as the resulting cost function in that case is not convex. To resolve this problem, a better cost function that is shown to have a global minima is used;

$$J(\theta) = \sum_{i=1}^{m} [-y^i \log(h_\theta(x^i)) - (1 - y^i) \log(1 - h_\theta(x^i))] \tag{5.7}$$

and the gradient of the cost is a vector of the same length as θ where the j^{th} element for $(j = 1, 2..., n)$ is given by

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} [((h_\theta(x^i)) - y^i)(x_j^i)] \tag{5.8}$$

The cost function in Equation 6 captures the notion of penalizing the learning algorithm when it makes a wrong prediction. Consider the case when $y = 1$ in the learning example and the hypothesis outputs $h_\theta(x) = 0$, thereby making a wrong prediction. Therefore, the term $-\log(h_\theta(x)$ in the cost function will be $-\log(0)$, which approaches infinity. However, contrary to this, if the hypothesis produces $h_\theta(x) = 1$, which is equal to the original $y = 1$ label, then the cost drops down to zero. The same reasoning can be applied to show the desired behavior of the cost function when $y = 0$.

On running the algorithm, on a subset of the data a threshold parameter vector θ is obtained which minimizes the above mentioned cost function $J(\theta)$. All such threshold parameters obtained from different subsets of data form the threshold parameter matrix Θ.

5.4.2 Feature selection

The choice of features in any machine learning algorithm play a pivotal role in the overall performance and accuracy of learning. We carefully selected the following features to use:

1. Time since last contact, t_l;
2. Node's timer value for the node in consideration, $v_i(v_k)$; and
3. Contacting node's timer value for the node in consideration, $v_j(v_k)$.

The rationale for using $v_i(v_k)$ and $v_j(v_k)$ is the fact that the difference of these two values gives us an indication whether the node in consideration is traveling towards or away from v_i. If v_i encounters a contacting node v_j which has a lower timer value for the node in consideration v_k, then intuitively v_i should be moving closer to v_k. However, if the v_j's timer is higher, then for most of the times an opposite inference can be made. We also make use of t_l, though this feature is not fed into logistic regression directly. We primarily use it to filter the data and run the algorithm on a subset of the data. It has been observed that the difference, $v_i(v_k) - v_j(v_k)$, which forms the column values x_1, contains more accurate information when t_l was small. This fact is also intuitive because with the passage of time the information becomes stale and less reliable.

5.5 Real-time estimation of directional mobility

5.5.1 Estimation at NOIs

As the nodes in the network move around and come into contact with each other opportunistically, they exchange information about their timers and update **MT** and **RT**. When a node $v_i \in P$ wishes to estimate the relative directional mobilities of the rest of the nodes in the network it first calculates the difference of arrays,

$$\mathbf{\Delta T_i} = \mathbf{MT_i} - \mathbf{RT_i} \tag{5.9}$$

that it maintains. The subscript i denotes that the above expression is evaluated for the i^{th} node. The node then checks its t_l value, i.e., how long ago did v_i make its last contact with some other node in the network. Based on the current t_l value, v_i chooses the corresponding t_l^{th} row (θ^T) from the threshold parameter matrix Θ. Now if v_i wishes to make a prediction about the direction of motion for node v_j, it calculates the following:

$$z = \theta^T \times \begin{bmatrix} 1 \\ \mathbf{\Delta T_i[j]} \end{bmatrix} \tag{5.10}$$

The index j fetches the $\mathbf{\Delta T_i}$ value for v_j. Subsequently, this z value is used in Equation (5.6) to get $g(z)$. If $g(z)$ is greater than or equal to zero, node v_i estimates the direction for v_j as approaching closer, whereas, $g(z)$ less than zero would mean otherwise. MOEME repeats the above mentioned procedure for each $v_i \in V - \{v_i\}$.

5.5.2 Time complexity and scalability

MOEME is extremely lightweight in terms of computation. The processing that a node $v_i \in P$ has to do to estimate the direction of motion of another node $v_j \in V - \{v_i\}$, comprises a fixed number of computational steps. The first step is simply taking the difference of two values to obtain a real number: $\mathbf{\Delta T_i[j]}$. The computation in Equation (5.10) is a multiplication of two 2×1 vectors, where the elements of the vectors are in $\mathbb{R}$. Finally, the computation of Equation (5.6) is also a fixed number of arithmetic operations. In order for a node to compute the direction for all the other nodes, the above mentioned steps shall be repeated $n - 1$ times, where n is the number of nodes in the environment. Therefore, the time complexity of MOEME for real-time estimation of directional mobilities is $O(n)$.

The machine learning algorithm we use in the chapter is lightweight and is essentially an offline training algorithm, which can run on a server machine. Therefore it does not affect real-time performance of MOEME. The machine learning algorithm we use is fairly quick and is able to train a data set with e.g., $18 \times 18 \times 90 \times 2 = 58320$ ($nodes \times nodes \times trace\_duration\_in\_minutes \times 60/30$) rows in under 3 minutes on a 2.4 GHz processor and 2 GB RAM, to produce a resultant threshold parameter matrix.

This analysis shows that MOEME can run with blazing fast performance on present day smart phones, that usually have processor speeds on the order of GHz. Therefore, MOEME is scalable to mobile environments consisting of nodes on the order of hundreds.

5.6 Direction estimation with orientation sensor

Thus far we have described a methodology to predict the directional mobility and count of nodes in a spatiotemporal region. However, the directional mobility is limited to estimating whether the nodes in question are moving towards or away from a node of interest. In this section, we detail the use of the orientation sensor on smartphone devices to get additional information about the angle of movement of a node.

5.6.1 Local orientation readings on a device

Modern smartphones are equipped with orientation sensors. For example, an Android device can make use of *getOrientation* in the *SensorManager* API [92] to get information about its orientation. In this chapter, every node v_i maintains a local orientation reading, which is essentially its own *bearing* value measured with respect to the magnetic north. We use the bearing values rounded to the nearest integer and are in the range $[0, 360)$. The value 0 corresponds to the the true magnetic north and the bearing increases when the device rotates eastward (clockwise). Figure 5.3 depicts the orientation coordinate system used in this chapter.

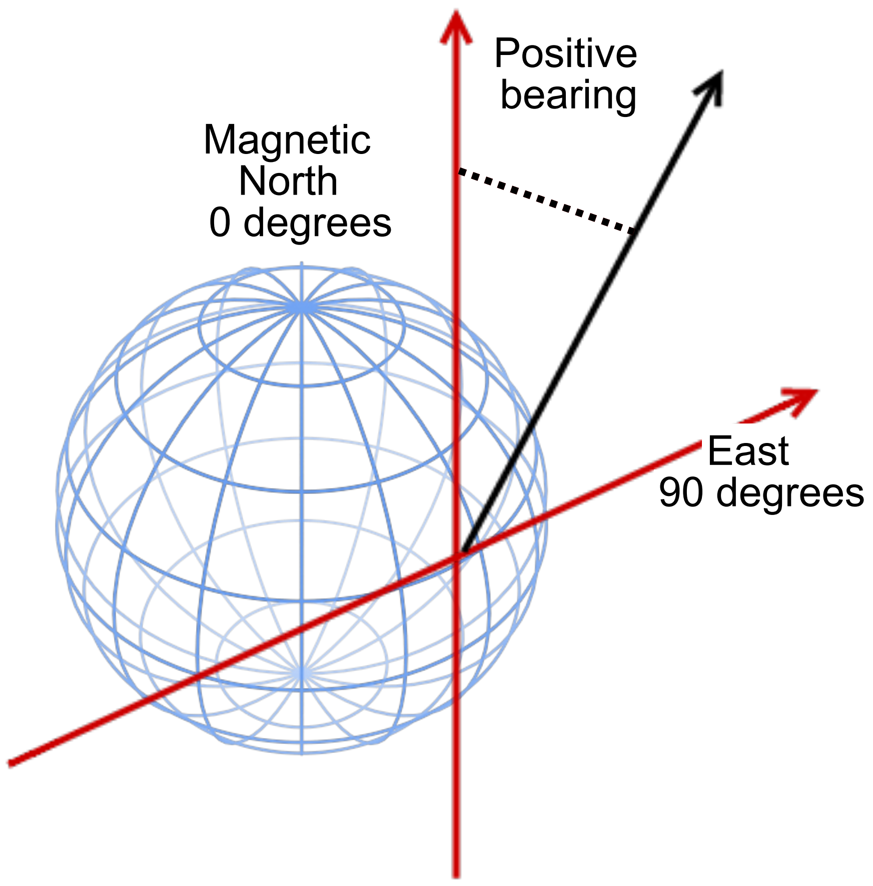


Figure 5.3: Orientation coordinate system

5.6.2 Sharing of orientation information

In a mobile environment the nodes move and often make contact with other nodes when they come within each other's radio communication range. A contact is an event where the nodes may exchange relevant information that can potentially help in achieving some goal. In this chapter, we use the notation $b_{v_i}(v_j)$ to denote the bearing value that node v_i maintains for node v_j. In other words, it is the absolute orientation of node v_j, perceived by node v_i. When the two nodes v_i and v_j come within each other's communication range, they inform the meeting node about their own orientation, i.e., the local bearing reading $b_{v_i}(v_i)$ at v_i is given to node v_j and vice versa. Moreover, much like the discussion in Section 5.3.5, each node maintains orientation information not just about itself or the meeting node, but about every other node in the network.

Initially, every node $v_i \in V$ in the network starts by keeping track of its own local bearing value using the orientation sensor. The bearing values for all other nodes at this node are initialized to -1, which means they are irrelevant and cannot be trusted. During the course of time, when the nodes move and a node $v_i \in V$ makes contact with $v_j \in V$, then the node v_j updates the orientation values according to the following bearing update rule:

$\forall v_k \neq v_j : t_{v_i}(v_k) < t_{v_j}(v_k) - t_{av}$, set,

$$b_{v_j}(v_k) = b_{v_i}(v_k) \tag{5.11}$$

The above update rule means that a node will prefer more recent perspective of orientation values that may be brought to it by other nodes in the network.

5.6.3 Prediction criterion and features

As time progresses, the nodes in the network share timer and orientation information among themselves, it is of interest to know whether the orientation information they maintain about other nodes is likely to be less accurate and hence a lower trust value. Intuitively, if the orientation information was updated long ago, then it should not be trusted. However, we relax the prediction criterion and instead of predicting the exact orientation angle of the nodes, we will tolerate the error within bounds.

In order to understand the scope of prediction consider the following function,

$$Q(x) = \begin{cases} 1 & if \quad 0 \leq x < 90 \\ 2 & if \quad 90 \leq x < 180 \\ 3 & if \quad 180 \leq x < 270 \\ 4 & if \quad 270 \leq x < 360 \end{cases}$$

that maps the integers between the range $[0, 360)$ to four different quadrants(divisions). Now, suppose the predicted bearing value at a node v_i for v_j is $b_{predict}(v_j)$ and the actual bearing value of v_j is $b_{actual}(v_j)$, at some time. Then, if $Q(b_{predict}(v_j)) = Q(b_{actual}(v_j))$, then we term the prediction as correct. In other words, it means if the predicted value and the actual value fall in the same division then the prediction is deemed correct. Note that, a different prediction criterion can easily be chosen by using a function $Q(x)$ with a different (smaller or greater) number of divisions. In this chapter, we have chosen four divisions as they naturally correspond to the quadrants. Moreover, note that it is not necessarily a very tolerant criterion, as the bearing values close to the boundaries i.e., 0, 90, 270 etc. may deviate only by a few degrees, and result in a wrong prediction.

To make predictions about the orientation of another node v_j, a node v_i uses the following two features as input to the sigmoid function:

1. Timer value for node v_j; and
2. Bearing value for node v_j.

While training the data set the actual bearing values of the nodes are plotted against the aforementioned features and a decision boundary is computed. The decision boundary can later be used to make predictions about the orientation of the nodes in real-time. Figure 5.4 shows one such data set at a node that was obtained during the training process. It shows whether the orientation values at a node fall within the same division. It can be seen that for higher timer values the actual value of the bearing is likely to move out of the same quadrant/division.

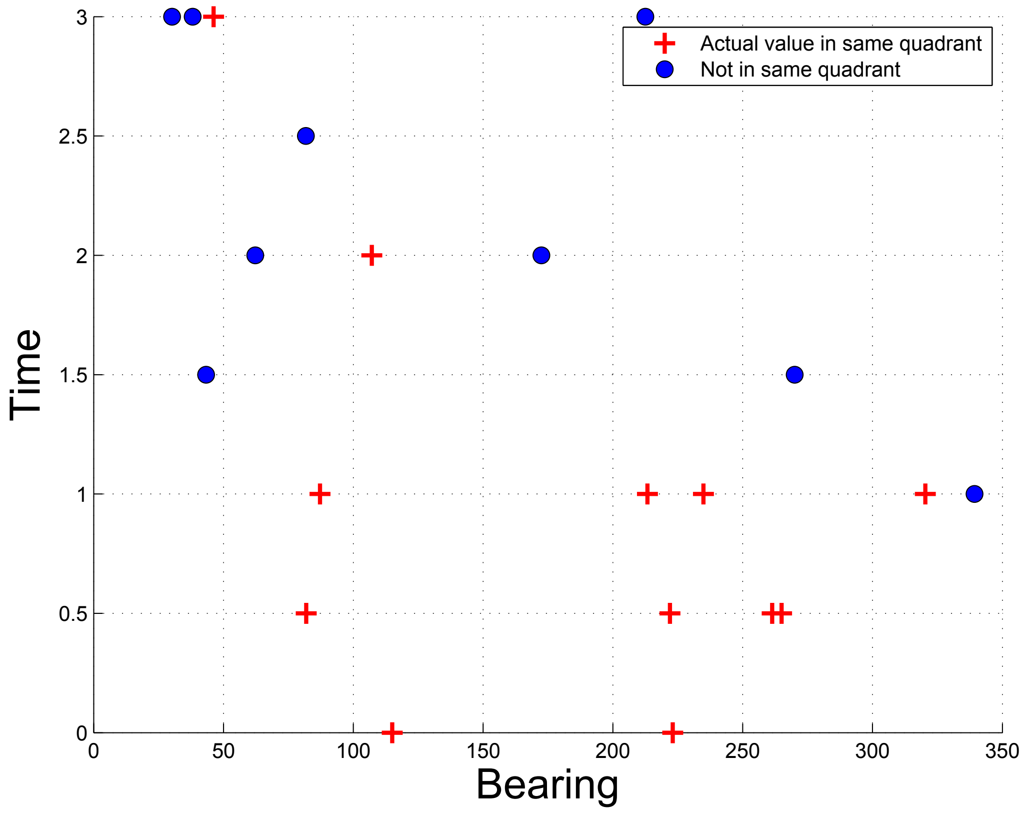


Figure 5.4: Data at an arbitrary node and time showing whether the orientation values fall within the same quadrant/division

5.7 Implementation

The architecture depicted in Figure 5.5 can be broken down into three major parts, i.e. Wi-Fi Direct, Network and Estimation, wherein each module is responsible to carry out a specific group of related tasks. MOEME is implemented and tested on Google Galaxy Nexus phones with a 1.2 GHz dual-core ARM Cortex-A9 CPU and 1 GB of memory. The phones are powered by Android 4.3 Jelly Bean.

5.7.1 Wi-Fi Direct module

Wi-Fi Direct lets mobile devices connect with each other directly without the need of an intermediate access point to over a 100m. The technology is available in Android versions greater than 4.0 and is supported by the above mentioned phones we use for implementing MOEME.

As part of MOEME the Wi-Fi Direct module is responsible for carrying out tasks related to establishing the connectivity with other nearby devices. Wi-Fi manager first tries to discover nearby peers by running a separate thread of execution. The interval between successive discovery initiations can be tweaked based on the environment, where a crowded environment would call for discovering often. Once the discovery returns a list of nearby devices, MOEME tries to connect to up to four available devices in the surrounding. This limitation on the design is placed to keep the implementation more responsive and agile. However, this does not mean MOEME misses out a chance to connect to other devices. Once the timer values are exchanged between the current set of connected devices, MOEME moves forward and connects to a set of mutually exclusive devices the next time around. Once there are no new devices to exchange timers with, a device cycles back to connect with the first set of nearby peers, if any or all of them are still available.

The information about connection setup or a disconnect is relayed via Wi-Fi Direct broadcast manager over to client/server maker. As the name suggests this sub-module is responsible for opening up server or client sockets depending on whether the connection earlier resulted in making a device a 'group owner' or not respectively. A group owner essentially acts as a hub, thereby routing messages among a peer to peer group. Therefore, a group owner is responsible for running and terminating several worker threads, each of which talks with the respective client device. On the contrary, each of the client devices simply connect to the group owner and synchronize their local timer values.

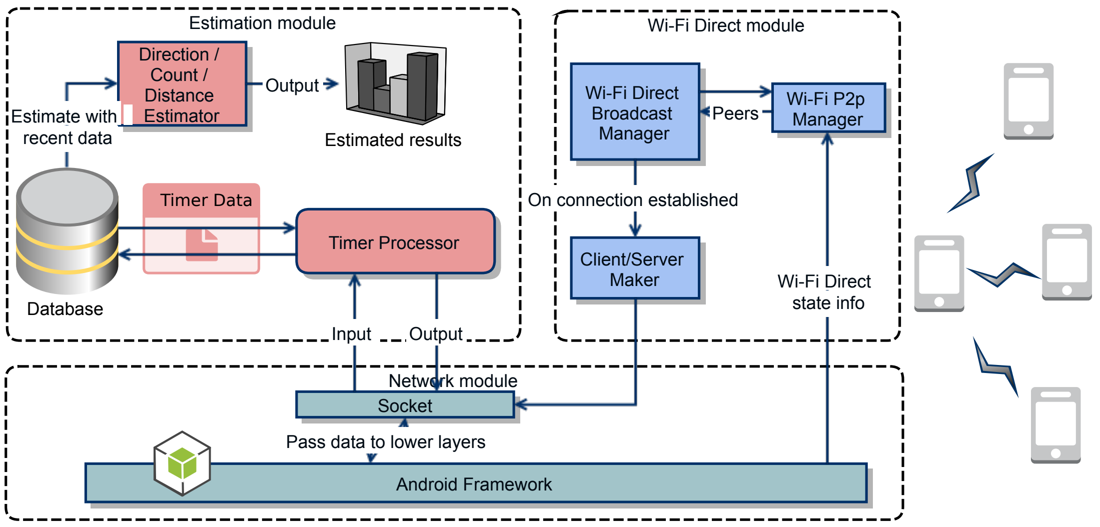


Figure 5.5: Architecture of MOEME implementation on an Android framework

5.7.2 Network module

The network module in Figure 5.5 shows the lower level networking layer. Sockets are essentially pipes between the estimation module and the Android framework, that carry streams of information wrapped around by conventional buffered readers and writers. The Android framework takes care of dealing with the low level networking i.e., sending and receiving the actual bits of information. Apart from this, the framework also monitors the state of several other ongoing processes, such as, peer listening and connection calls.

5.7.3 Estimation module

The primary purpose of MOEME is to estimate mobilities of users in the surrounding environment. The estimation module, contains implementation of the logic described earlier in timer update rule and real-time estimation of orientation/directional user mobilities. First, timer processor receives new information about a peer's timer values. **MT** and **RT** are updated along with the orientation information, in an SQLite database. Subsequently, a node's own timer values and local orientation information are pulled out from the database and transferred over to the connected peer. Subsequently, the timer values and orientation information are updated according to the timer update rule 5.2 and bearing update rule 5.11 respectively.

Finally, the mobility estimator reads the most current view of the network and estimates the directional mobilities and orientation of the users it has information about. The results of the estimation are then presented to the user in the form of Android activities. Which are basically a way for the user to interact with an Android UI. Android makes it easy for several applications to communicate with each other. Therefore, it is possible for other applications to use services of MOEME, given the correct privileges are in place.

5.8 Simulation and Analysis

Performance of prediction is evaluated for estimating direction, distance, and count of nodes within a certain region. The performance varies when there has been a long time since last contact (represented by time range) and when nodes are located physically further apart (represented by distance range). Extensive simulations are run on real as well as synthetic mobility traces. The real mobility trace has been collected from participants that carry GPS receivers which log position at 30 second intervals. These traces are collected in five different environments, but we show representative results from a State Fair [93] in Figures 5.6 and 5.7. State Fair's area is 500m x 500m and the nodes have typical walking speeds with recurrent pauses at different stalls.

In order to make a comparison with suitable number of nodes, track logs from each day are considered to be that of a separate user. These logs have durations from around one to twelve hours each day. We truncate all logs to 90 minutes during which 18 user devices record their location. As the trace logs have different durations, ranging from one hour to twelve hours, the logs were clipped to make the analysis consistent, i.e., for the first 90 minutes. For longer than 90 minutes, the number of nodes that log information for a longer duration decreases sharply. So we decided to use 90 minute duration with 18 logs available (instead of e.g. 4 hours duration with only 7 logs available for that duration).

A similar preprocessing steps were performed on the other four real-world mobility traces. We show the orientation prediction accuracy of MOEME for all of them in Figure 5.8. Table 5.1 shows the number of nodes along with the duration to which each trace was clipped for uniformity.

Table 5.1: Real-world mobility traces used in this chapter to show orientation prediction accuracy

Trace	**Number of nodes**	**Duration (minutes)**
State Fair	18	90
Orlando	41	131
New York	39	74
NCSU	35	103
KAIST	92	253

Synthetic mobility traces are generated using SLAW mobility model [94] with 20 nodes. Length of each step, and pause duration is different for all nodes throughout the trace duration, where we only specify the exponent of distribution from which a random value is selected for every node at every step. The environment is made further heterogeneous by changing velocity, minimum/maximum pause durations, and minimum step length for two classes of nodes i.e., 10 slow and 10 fast moving nodes. Details about trace settings are provided in Table 5.2.

Table 5.2: SLAW mobility trace used for simulation

Description	**slow**	**fast**
Exponent of step length distribution	1.6	
Exponent of pause time distribution	0.6	
Hurst Parameter	0.75	
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	

5.8.1 Simulation setup

Figures 5.6 and 5.7 show the average of results at all nodes at different times. Since, the accuracy results belong to a Bernoulli distribution, the 99% confidence interval are extremely close to the average values due to large number of samples (greater than 2000). Therefore, the confidence intervals are not plotted. Parameters used for prediction are $t_{av} = 3$min and the linear fit line to compute timer thresholds based on distance threshold is distance $= \sqrt{t_{v_i}(v_j)} + 28$. This line is based on linear fit between square root of timers [14] and actual distances between nodes in the state fair trace. Thus, to estimate if distance between two nodes is less than a specific value, we check the node's timer if it is less than the value satisfied by the above equation[1]. We make predictions for three measures:

1. Direction: relative direction of users with respect to NOI;
2. Distance: distance of a user from NOI to be less than a specific threshold,

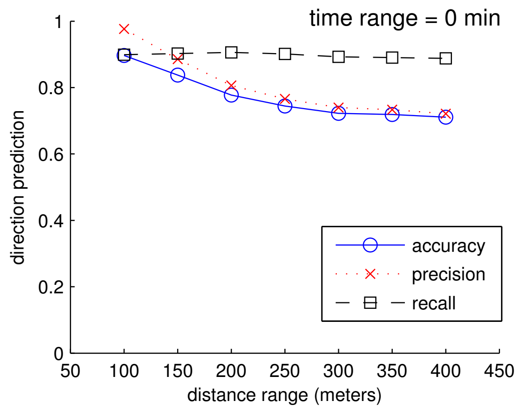
[1]The unit of distance and time is in meters and minutes respectively

3. Count: number of users whose distance from NOI is less than a specific threshold; and
4. Orientation: the absolute bearing value of users predicted at a NOI.

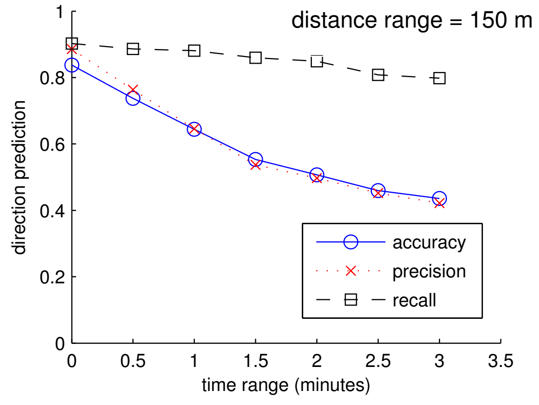
The prediction of above measures is analyzed against two variables: distance range and time range. For example, when distance range is 200 m, it means that the prediction is made for a selection of users that are within 200m of any NOI. When time range is 10 min, it means that the prediction is made for a selection of users whose timer value has been updated at the NOI within previous 10 min. Thus, a smaller time range means that prediction is made for a selection of users about whom some information (the time value) has been updated recently. Intuitively, prediction for small values of time range and distance is better because it means that the user is situated nearby and the information (time value) about the node has been updated recently.

5.8.2 Direction estimation

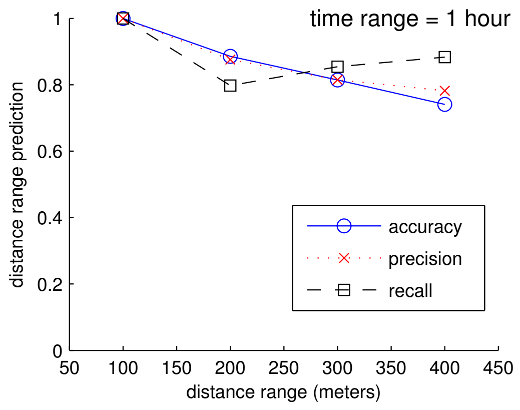
From the logistic regression model, we find different values of θ specific to state fair trace for different values of time since last contact. In order to generalize, we find that use of $\theta_0 = 0$ gives good results for both the state fair as well as synthetic mobility models. This simplifies to simply using **MT–RT**, and inferring the movement of user to be moving close to NOI for positive difference in timers. Figure 5.6c shows the precision, recall and accuracy for all users (i.e. the results are average by considering each user to be the NOI) at different distance ranges. Precision and recall show the efficiency of prediction when a specific user is moving closer to the NOI. The recall is greater than 90% at all distance ranges, however, precision and accuracy drop from 90% to 75% when the distance range increases from 100m to 300m. This is because, for users that are further away, it takes longer for new information to reach a NOI. However, even a 75% accuracy is extremely



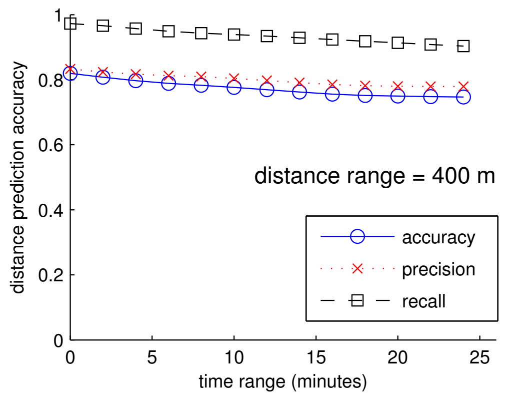
(a) Prediction of direction of nodes that are close is better



(b) Prediction of direction of nodes decreases sharply when no new information is available in



(c) Prediction of distance of nodes that are close is better



(d) Prediction of distance of nodes decreases moderately when no new information is available in past 10-20 minutes

Figure 5.6: Direction and distance estimation

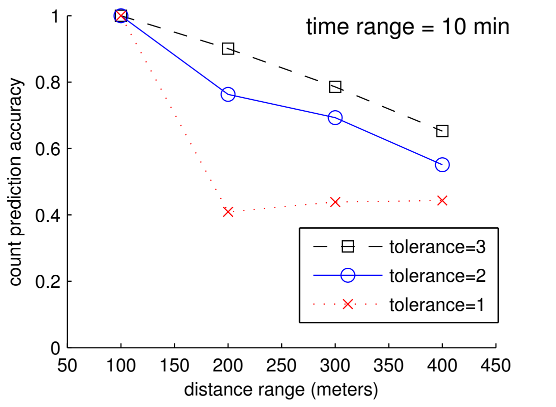
useful as there does not exist any other mechanism to estimate direction of users in physical proximity by using the opportunistic transfer of information and without any GPS device.

In contrast to prediction robustness against users that are located further away, time since last contact (represented by time range) sharply decreases the precision and accuracy. This is reflected in Figure 5.6d, where prediction accuracy drops from 90% to 50% in just over 2.5 minutes. This is because there is almost zero correlation in direction of user movement between two instants of time separated by more than 3 minutes. We believe this is a special case of state fair traces and in other environments with directional flow, the prediction accuracy will drop more gradually. However, this effect needs to be tested based on collection of real mobility data from additional environments with directional flows.

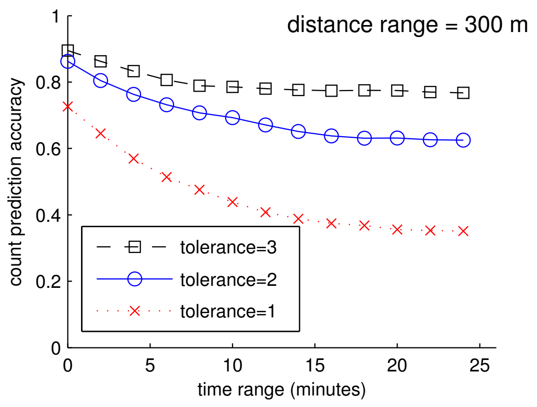
5.8.3 Distance estimation

Since, the direction estimation is sensitive to time range, we estimate the distance of users in proximity. Thus, we find if a particular user is within a specific distance of NOI where this specific distance is given by the distance range. Figure 5.6a shows that distance prediction (precision and accuracy) drops from 100% at 100m (100m is also the transmission range of NOI) to 80% at 300m whereas recall initially decreases to 80% but then increases back to 90% at distance range of 400m. The drop in precision and accuracy is intuitive but the increase in recall is an artifact of specific user mobility in the state fair trace and is a result of linear fit of: i) distance with square root of temporal distance; and ii) the space constraint of approximately 500x500m in state fair. Figure 5.6b shows that distance prediction is only moderately affected by the time range i.e. the accuracy drops from 0.80 to 0.77 when time range increases from 0 min to 25min. The reason is that even though users change direction a lot more rapidly (very likely in 3 min [Figure 5.6d]), they do not move out of a region that quickly. Thus a user within 400m of NOI is likely to remain in the region for another 20 minutes even though it may keep on changing directions more quickly.

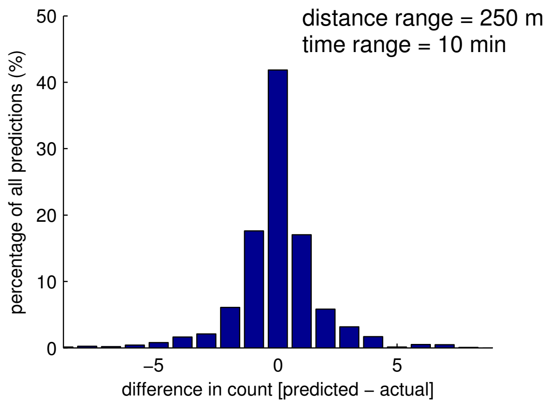
MOEME scheme performs better in dense environments. With higher ranges, the latency for receiving timer values is larger in environments such as the ones we are targeting (parks, malls, streets etc.). With longer delays there is a higher probability that the user would change direction thus making it difficult to use the prediction made by MOEME. This is an inherent characteristic of human mobility in some dynamic environments. MOEME would perform better with high range if users do not change direction frequently.



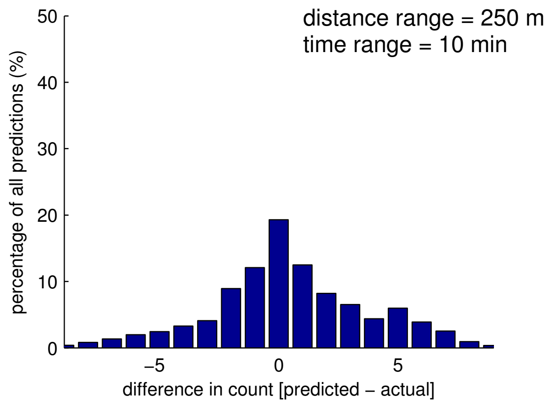
(a) Prediction of count of nodes that are close is better



(b) Prediction of count of nodes decreases moderately when no new information is available in past 10-20 minutes



(c) 80% of count predictions are within an error of a single count in state fair



(d) 80% of count predictions are within an error of a single count in SLAW Mobility Model

Figure 5.7: Count prediction

5.8.4 Count estimation

In addition to the distance estimate, we also count the number of users within a certain region as defined by the distance range. Since, this count may not be exact, we test the prediction with different levels of tolerance i.e. a tolerance of 1 means that actual count and predicted count can differ at most by one. Therefore, prediction accuracy is higher when tolerance is 3 as opposed to 1. This is also reflected in Figure 5.7a. Similar to distance estimation, the count estimation is more robust to time ranges in comparison with distance ranges. When tolerance is 1, the count prediction drops to 50% at a distance range of 200m - but with tolerance equal to 2, the accuracy is close to 80% as shown in Figure 5.7a.

Figure 5.7b shows the effect of time ranges. For tolerance equal to 1, the accuracy drops below 50% at 15min but at tolerance equal to 3, the accuracy stays above 80% for up to time range of 25% which is quite good.

In order to analyze the error in count more deeply, Figure 5.7c and Figure 5.7d show a histogram of different in actual and predicted count. As seen from the figure, the tolerance of 3 enables an accuracy of up to 80% in state fair and 70% in SLAW mobility model. The error in SLAW mobility is more dispersed as it is a random movement model and there is much less correlation in directions of user movement.

5.8.5 Orientation estimation

The orientation of users is predicted at NOIs using the trained logistic regression model. The plot shown in Figure 5.8 is generated for users in five different real-world mobility traces. The simulation is run a 1000 times, and in each run arbitrary NOIs are chosen that make predictions about the orientation of other nodes for the entire duration of the trace. It is observed that the prediction accuracy is 100% accurate when the timer information is just exchanged. This is because, the bearing values exchanged at the time of contact are fresh and depict accurate information about the actual orientation of nodes. However, the prediction accuracy drops as the temporal distance increases between the nodes. This means that no new information about the bearing values of the nodes in question is known,

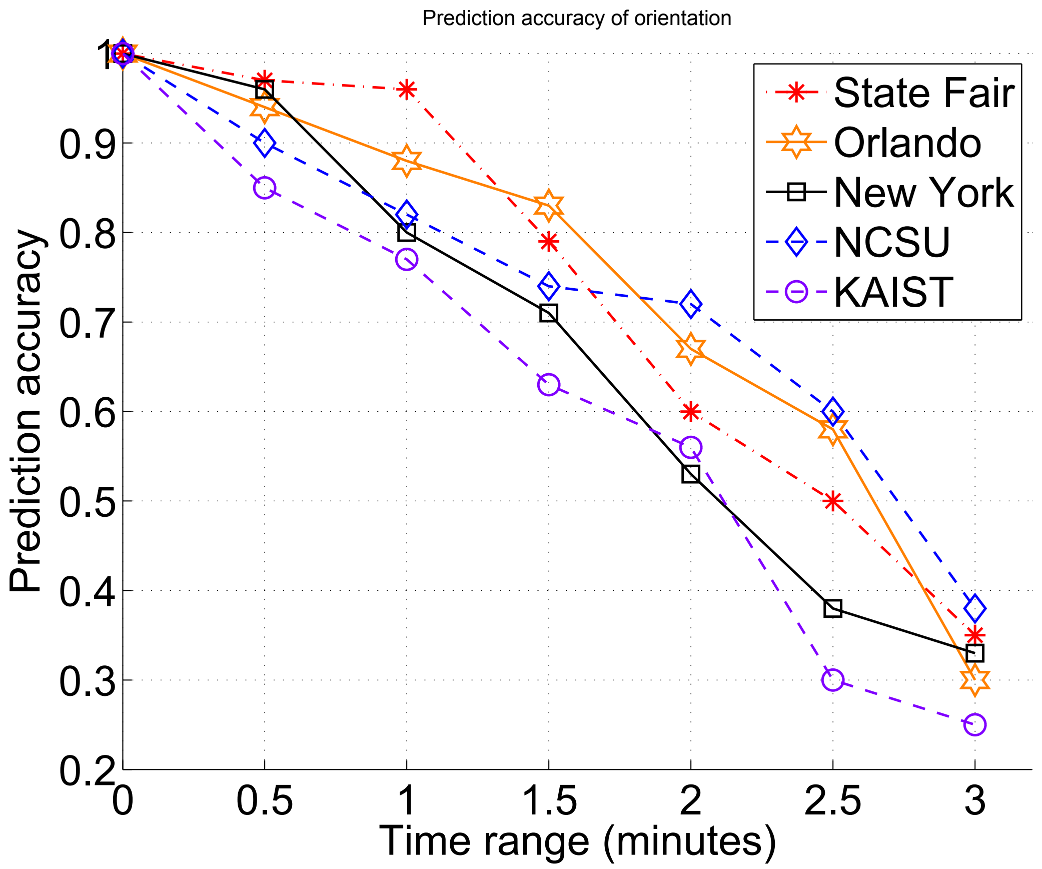


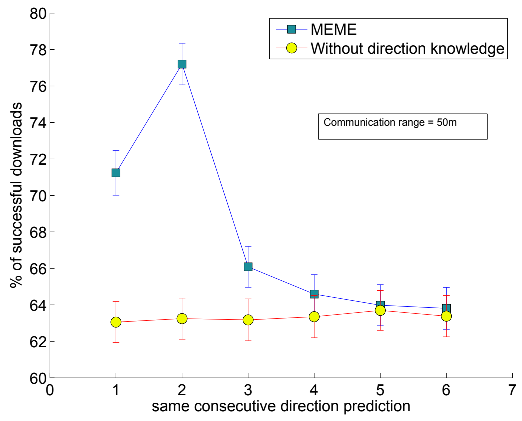
Figure 5.8: Prediction accuracy of orientation against timer values. The accuracy is plotted for the five different real-world mobility traces.

hence the prediction accuracy drops sharply. KAIST trace shows the worst prediction accuracy, and we believe it is because it has the maximum number of nodes for which the timer and bearing values are not updated soon enough.

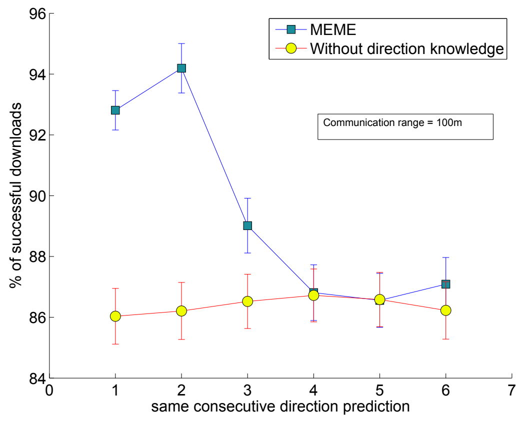
5.8.6 Successful downloads in a video sharing application

In order to show an application of MOEME, a video sharing service is simulated in a mobile environment. At the start of the experiment, a cluster of three nodes is chosen, wherein each node downloads a chunk of video from the cellular network and maintains it for future downloads over speedy and free local Wi-Fi links. Other mobile nodes in the environment can download individual chunks of original video from the provider nodes if they come within communication range and the Wi-Fi link is not broken for the entire duration of the download.

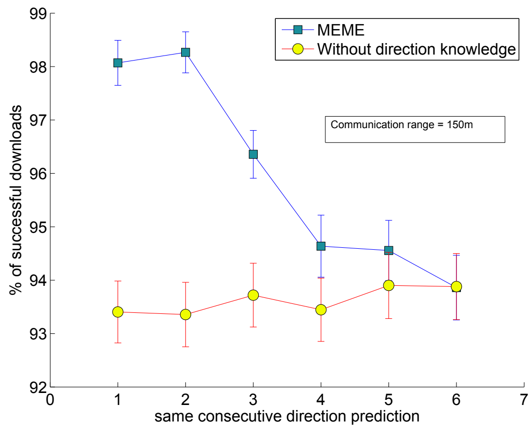
As MOEME powered devices can estimate directional mobilities, a simple heuristic is used to improve performance. The provider nodes in the cluster estimate the direction of movement from each other and keep a memory of past λ number of *same* consecutive direction predictions. Figure 5.9 shows λ varying from 1 to 6 on the x-axis. Provider nodes continuously monitor each other's movement. Suppose a provider node v_i is found to move away from the cluster for λ consecutive predictions, a suitable current non provider node moving towards the cluster for λ consecutive predictions is identified to take over from v_i as a new provider. Figure 5.9 plot the average successful percentage of downloads performed for 2000 runs of simulation. It is observed that, using a simple heuristic based on MOEME, for a communication range of 50 m, average number of downloads served is improved by more than 16%. The gain of MOEME over methods that do not use directional information is most noticeable for short communication ranges. However, it is observed, that the overall performance of servicing video downloads improves for longer communication ranges. This is because, link failures in such an environment are relatively less. Moreover, it is observed that with $\lambda = 2$, MOEME performs best and the performance asymptotically degrades with higher λ. This is because, it is a stricter condition and finding nodes that keep their relative directional movement constant over longer period of times are hard to find in a dynamic mobile environment.



(a) Gain with MOEME is most pronounced for a short communication range (50 m)



(b) Performance with MOEME peaks at 94 % for 100 m communication range



(c) Long communication range (150 m) improves overall success rate

Figure 5.9: Percentage of successful downloads for different communication ranges. MOEME performs better in all three cases

5.9 Summary

This chapter presents the first fully-distributed and real-time scheme to estimate direction, distance and users in the proximate environment as well as the count of users within a specific distance of any node of interest (NOI). The prediction of direction utilizes a logistic regression model and gives more than 77% accuracy for nodes within 200 m of the NOI whenever the NOI's timer is updated for a particular user. The estimates of distance and count are more robust even when no new contact has occurred in the past 10 to 15 minutes and prediction accuracy is greater than 80% for users with 300m of the NOI. The proposed scheme demonstrates similar performance in terms of prediction accuracy for real and synthetic mobility traces. It is also shown that MOEME can be employed to enhance the percentage of successful downloads by over 14 % for a video sharing service in a mobile environment. This opens up possibilities for intelligent resource allocation and event scheduling in various other multimedia applications.

CHAPTER 6

Conclusion

6.1 Summary

In this dissertation we have presented solutions to three important problems pertaining participatory environments, that serve as building blocks to achieve richer collaboration and data transfers. Participatory environments are highly dynamic and thus, there is usually no end to end connection between source and destination nodes. In this dissertation we presented three novel schemes:

1. A novel scheme for predicting contact volume during opportunistic contacts (PCV);
2. A method for computing paths with combined optimal stability and capacity (COSC) in opportunistic networks; and
3. An algorithm for mobility and orientation estimation in mobile environments (MOEME).

The PCV scheme serves to find the maximum allowable data transfer between a pair of opportunistically meeting nodes, whereas COSC utilizes PCV, and determines the combined optimal stable and capacity paths in participatory environments. MOEME is a distributed and scalable scheme that can be employed to predict users' mobilities in a dynamic environment. MOEME also has the capability to detect the orientation of mobile nodes on top of directional mobilities.

PCV is a lightweight scheme with a flexible architecture. To demonstrate its flexible structure, we show how PCV uses data rate profile learnt with DatPro Application. We have also presented a linear time algorithm to compute the contact volume between pairs of nodes in an environment. Extensive simulation results demonstrate the efficacy of our approach in terms of reducing bandwidth wastage. Moreover, it is shown that the effect of false negatives is minimal in PCV.

COSC builds on top of PCV, wherein two objectives are taken into account for improving data transfers between source and destination pairs. COSC takes into account both capacity and stability to come up with a utility function that can be solved optimally using a dynamic programming approach. However, as the optimal paths computed may not be applicable at run-time due to the dynamic nature of the environment, COSC has the ability to fall back to more traditional routing mechanisms. Moreover, the results obtained in this study can be used to evaluate the suitability of applications in different user movement scenarios.

MOEME can be employed to estimate the directional mobilities and orientation of participating nodes in a dynamic environment. It is particularly useful for indoor environments where GPS signals are weak and prone to large errors. MOEME is fully distributed and uses logistic regression to classify directional mobilities and orientations. It is also shown that MOEME can be employed to enhance the percentage of successful downloads by over 14 % for a video sharing service in a mobile environment. This opens up possibilities for intelligent resource allocation and event scheduling in various other multimedia applications. All the simulation results shown in this dissertation are conducted on a variety of synthetic and real-world mobility traces that encompass scenarios with a wide range of statistical properties.

6.2 Broader applications

The work presented in this dissertation has a wide variety of applications. Consider an emergency evacuation scenario in an indoor movie theatre. The goal of any successful evacuation plan is to guide people out of the building as quickly and safely as possible.

However, it is observed during evacuations that: 1) humans tend to crowd only a subset of all available exits; and 2) people may not be aware of all available exits. The solutions presented in this dissertation are directly applicable to the described scenario. By equipping exits with smart and appropriate technologies, MOEME can be employed to estimate people's directions and orientations during evacuation. Whenever, certain exits tend to get crowded, a subset of users can be informed (via opportunistic message exchange) of other available exits in a building. Furthermore, COSC can be employed to find suitable cost effective paths for relaying these messages.

PCV and COSC, are also worthwhile options for gaming and entertainment applications. As mobile phones are getting powerful everyday, programmers and developers are creating fully functional 3D games, utilizing every computing cycle graphics cards have to offer. Though, such graphics intensive games promise a fine user experience on standalone smartphones, they may not perform very well in a p2p multiplayer environment. Thus, it is important to make communication and collaboration among devices more efficient to deliver a better experience for multiplayer gaming applications. Moreover, the proposed schemes have applications in several other real life situations including video sharing, participatory sensing and crowd sourcing.

6.3 Future work

6.3.1 Contact volumes over multiple devices

These days it is common to see people carrying more than one smart device. Currently, PCV takes into account a single device per person, and models the interaction between two such people. PCV can be extended by developing prediction models for people carrying multiple devices, where devices carried by one person are always within the communication range of each other.

6.3.2 Path sharing in COSC

COSC is a scheme to find a path set that has combined optimal stability and capacity over a time varying graph. However, COSC does not allow users to share links among several paths. For example, consider two source destination pairs s_1, d_1 and s_2, d_2. The path set over a TVG between s_1, d_1 may contain an edge e_i in the i-th timestamp. Currently, COSC does not find optimal cost paths between s_2, d_2 that may partially utilize the edge e_i. It will be interesting to determine the performance of path sets that share edges with each other.

6.3.3 UAVs & drones

The work presented in this dissertation is primarily tested on real-world human mobility traces and synthetic human mobility models. This work can be extended by testing it on other mobility models, such as those in Unmanned Aerial Vehicles (UAVs) and drones.

6.3.4 Vehicular networks

The work presented in this dissertation can be employed in vehicular networks. Though the trajectories followed by vehicles have more patterns and relatively simpler than human mobility, there is nonetheless an overlap. Vehicles can also make use of the *store-forward* paradigm, especially while stopping at traffic signals. Is it possible to predict mobility and orientation of vehicles in real-time using MOEME? Can COSC still find combined optimal stable and capacity paths in a TVG, where vehicles are modeled as vertices?

REFERENCES

[1] L. Keller, A. Le, B. Cici, H. Seferoglu, C. Fragouli, and A. Markopoulou, "Microcast: Cooperative video streaming on smartphones," in *Proceedings of the 10th ACM International Conference on Mobile systems, applications, and Services (MOBISYS)*, 2012, pp. 57–70. Cited on pages 2 and 38.

[2] G. Ananthanarayanan, V. N. Padmanabhan, L. Ravindranath, and C. A. Thekkath, "Combine: leveraging the power of wireless peers through collaborative downloading," in *Proceedings of the 5th ACM International Conference on Mobile Systems, Applications and Services (MOBISYS)*, 2007, pp. 286–298. Cited on page 2.

[3] M. Stiemerling and S. Kiesel, "A system for peer-to-peer video streaming in resource constrained mobile environments," in *Proceedings of the 1st ACM Workshop on User-provided Networking: Challenges and Opportunities*, 2009, pp. 25–30. Cited on page 2.

[4] M. Ramadan, L. El Zein, and Z. Dawy, "Implementation and evaluation of cooperative video streaming for mobile devices," in *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*. IEEE, 2008, pp. 1–5. Cited on page 2.

[5] P. Sharma, S.-J. Lee, J. Brassil, and K. G. Shin, "Handheld routers: Intelligent bandwidth aggregation for mobile collaborative communities," in *Broadband Networks, 2004. BroadNets 2004. Proceedings. First International Conference on*. IEEE, 2004, pp. 537–547. Cited on page 2.

[6] A. Chaintreau, A. Mtibaa, L. Massoulie, and C. Diot, "The diameter of opportunistic mobile networks," in *Proceedings of the 2007 ACM CoNEXT conference*. ACM, 2007, p. 12. Cited on page 3.

[7] A. Passarella and M. Conti, "Characterising aggregate inter-contact times in heterogeneous opportunistic networks," in *NETWORKING 2011*. Springer, 2011, pp. 301–313. Cited on pages 3 and 40.

[8] P. Grindrod and D. J. Higham, "Evolving graphs: dynamical models, inverse problems and propagation," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, vol. 466, no. 2115, pp. 753–770, 2010. Cited on page 3.

[9] M. Conti, M. Mordacchini, A. Passarella, and L. Rozanova, "A semantic-based algorithm for data dissemination in opportunistic networks," in *Proceedings of the 7th International Workshop on Self-Organizing Systems (IWSOS)*, 2013. Cited on pages 3 and 13.

[10] A.-K. Pietilänen and C. Diot, "Dissemination in opportunistic social networks: the role of temporal communities," in *Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing*. ACM, 2012, pp. 165–174. Cited on pages 3 and 13.

[11] J. Whitbeck, M. Dias de Amorim, V. Conan, and J.-L. Guillaume, "Temporal reachability graphs," in *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 2012, pp. 377–388. Cited on page 3.

[12] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *Mobile Computing, IEEE Transactions on*, vol. 6, no. 6, pp. 606–620, 2007. Cited on pages 3 and 47.

[13] A. Vahdat, D. Becker, *et al.*, "Epidemic routing for partially connected ad hoc networks," Technical Report CS-200006, Duke University, Tech. Rep., 2000. Cited on pages 3, 11, and 64.

[14] U. Sadiq and M. Kumar, "Proximol: Proximity and mobility estimation for efficient forwarding in opportunistic networks," in *Proceedings of the 8th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, 2011, pp. 312–321. Cited on pages 4, 11, 14, 21, 64, 67, 69, and 86.

[15] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-coding based routing for opportunistic networks," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*. ACM, 2005, pp. 229–236. Cited on pages 4 and 64.

[16] P. Santi, "Routing in opportunistic networks," *Mobility Models for Next Generation Wireless Networks: AD HOC, Vehicular and Mesh Networks*, pp. 225–236, 2012. Cited on pages 4 and 64.

[17] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "Slaw: A new mobility model for human walks," in *IEEE INFOCOM*, 2009, pp. 855–863. Cited on pages 4, 29, 30, 33, and 47.

[18] C. Boldrini and A. Passarella, "Hcmm: Modelling spatial and temporal properties of human mobility driven by users social relationships," *Computer Communications*, vol. 33, no. 9, pp. 1056–1074, 2010. Cited on pages 4, 47, and 56.

[19] S. Qayyum, "DatPro: Data Rate Profiling Agent," https://github.com/shirazqayyum/WSN-APP, 2012, [Online; accessed 03/12/2013]. Cited on page 4.

[20] S. Qayyum, M. Shahriar, M. Kumar, and S. K. Das, "Pcv: Predicting contact volume for reliable and efficient data transfers in opportunistic networks," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*. IEEE, 2013, pp. 801–809. Cited on pages 5, 41, 43, 48, 53, 60, and 65.

[21] L. Vu, Q. Do, and K. Nahrstedt, "Jyotish: A novel framework for constructing predictive model of people movement from joint wifi/bluetooth trace," in *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*. IEEE, 2011, pp. 54–62. Cited on page 8.

[22] M. Marchini, M. Tortonesi, G. Benincasa, N. Suri, and C. Stefanelli, "Predicting peer interactions for opportunistic information dissemination protocols," in *IEEE Computers and Communications (ISCC)*, 2012, pp. 000 512–000 517. Cited on pages 8 and 19.

[23] Q. Yuan, I. Cardei, and J. Wu, "Predict and relay: an efficient routing in disruption-tolerant networks," in *Proceedings of the 10th ACM International Symposium on Mobile Ad hoc Networking and Computing*, 2009, pp. 95–104. Cited on page 8.

[24] H. Chowdhury, J. Lehtomäki, J.-P. Mäkelä, and S. Kota, "Data downloading on the sparse coverage-based wireless networks," *Journal of Electrical and Computer Engineering*, vol. 2010, p. 34, 2010. Cited on page 8.

[25] J. P. Neto, E. Nascimento, E. Mota, E. Cerqueira, P. Almeida, and R. Rojas, "A model for contact volume prediction in dtns," in *IEEE Symposium on Computers and Communications (ISCC)*, 2012, pp. 000 199–000 202. Cited on page 8.

[26] N. Banerjee, M. D. Corner, and B. N. Levine, "An energy-efficient architecture for dtn throwboxes," in *26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, 2007, pp. 776–784. Cited on page 8.

[27] K. Benkic, M. Malajner, P. Planinsic, and Z. Cucej, "Using rssi value for distance estimation in wireless sensor networks based on zigbee," in *Proceedings of the 15th IEEE International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2008, pp. 303–306. Cited on page 9.

[28] T. Hao, R. Zhou, and G. Xing, "Cobra: color barcode streaming for smartphone systems," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 85–98. Cited on page 9.

[29] S. D. Perli, N. Ahmed, and D. Katabi, "Pixnet: interference-free wireless links using lcd-camera pairs," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 2010, pp. 137–148. Cited on page 10.

[30] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Efficient routing in intermittently connected mobile networks: The single-copy case," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 1, pp. 63–76, 2008. Cited on pages 11 and 70.

[31] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Single-copy routing in intermittently connected mobile networks," in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*. IEEE, 2004, pp. 235–244. Cited on page 11.

[32] L. Yin and G. Cao, "Supporting cooperative caching in ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 1, pp. 77–89, 2006. Cited on page 11.

[33] A. Valera, W. K. G. Seah, and S. Rao, "Cooperative packet caching and shortest multipath routing in mobile ad hoc networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 1. IEEE, 2003, pp. 260–269. Cited on page 11.

[34] B. D. Higgins, J. Flinn, T. Giuli, B. Noble, C. Peplin, and D. Watson, "Informed mobile prefetching," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 155–168. Cited on page 11.

[35] B. B. Xuan, A. Ferreira, and A. Jarry, "Computing shortest, fastest, and foremost journeys in dynamic networks," *International Journal of Foundations of Computer Science*, vol. 14, no. 02, pp. 267–285, 2003. Cited on page 13.

[36] A. Casteigts, P. Flocchini, B. Mans, and N. Santoro, "Deterministic computations in time-varying graphs: Broadcasting under unstructured mobility," in *Theoretical Computer Science*. Springer, 2010, pp. 111–124. Cited on page 14.

[37] J. Ghosh, S. J. Philip, and C. Qiao, "Sociological orbit aware location approximation and routing (solar) in manet," *Ad Hoc Netw.*, vol. 5, no. 2, pp. 189–209, 2007. Cited on pages 14 and 67.

[38] S. Nelson, M. Bakht, and R. Kravets, "Encounter-based routing in dtns," in *Proc. IEEE INFOCOM*, 2009. Cited on page 14.

[39] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks," in *Proc. IEEE INFOCOM*, 2006. Cited on page 14.

[40] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks," in *Proc. ACM MobiHoc*, 2008. Cited on page 14.

[41] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proc. ACM MobiHoc*, 2007. Cited on page 14.

[42] 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 page 14.

[43] M. C. C. Boldrini and A. Passarella, "Contentplace: Social-aware data dissemination in opportunistic networks," in *Proc. MSWiM*, 2008. Cited on page 14.

[44] U. Sadiq, M. Kumar, A. Passarella, and M. Conti, "Modeling and simulation of service composition in opportunistic networks," in *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*. ACM, 2011, pp. 159–168. Cited on pages 14 and 67.

[45] A. Pal, "Localization algorithms in wireless sensor networks: Current approaches and future challenges," *Network Protocols and Algorithms*, vol. 2, no. 1, pp. 45–73, 2010. Cited on pages 14 and 16.

[46] "Radio Interferometric Localization," http://www.isis.vanderbilt.edu/projects/nest/selfloc.html, [Online; accessed 27-April-2013]. Cited on page 16.

[47] J. Whitbeck, M. De Amorim, and V. Conan, "Plausible mobility: inferring movement from contacts," *arXiv preprint arXiv:1001.3673*, 2010. Cited on pages 17 and 67.

[48] A. Mazzini, C. Stefanelli, M. Tortonesi, G. Benincasa, and N. Suri, "Disservice: Network state monitoring and prediction for opportunistic information dissemination in

tactical networks," in *IEEE MILITARY COMMUNICATIONS CONFERENCE (MILCOM)*, 2010, pp. 555–560. Cited on page 19.

[49] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding," Technical Report UCAM-CL-TR-617, University of Cambridge, Computer Laboratory, Tech. Rep., 2005. Cited on page 19.

[50] J. D. Gibson, *Mobile communications handbook*. CRC press, 2012, vol. 45. Cited on pages 23 and 48.

[51] V. Abhayawardhana, I. Wassell, D. Crosby, M. Sellars, and M. Brown, "Comparison of empirical propagation path loss models for fixed wireless access systems," in *IEEE Vehicular Technology Conference (VTC) 2005*, vol. 1, 2005, pp. 73–77. Cited on page 25.

[52] G. Strang, *Computational science and engineering*. Wellesley-Cambridge Press Wellesley, MA, 2007. Cited on pages 26 and 27.

[53] Android, "Wifi direct." Cited on pages 26, 44, and 57.

[54] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong, "CRAWDAD Trace," http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels/GPS/NC_State_Fair, July 2009. Cited on pages 29 and 30.

[55] I. Rhee, M. Shin, S. Hong, K. Lee, and S. K. . S. Chong, "CRAWDAD Trace," http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels/GPS/Disney_World, July 2009. Cited on pages 29 and 30.

[56] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong, "On the levy-walk nature of human mobility," *IEEE/ACM Transactions on Networking (TON)*, vol. 19, no. 3, pp. 630–643, 2011. Cited on pages 29, 30, and 31.

[57] Seongik, M. Hong, and Shin, "Truncated Levy Walk (TLW) generator," http://research.csc.ncsu.edu/netsrv/?q=content/

human-mobility-models-download-tlw-slaw, 2009, [Online]. Cited on page 31.

[58] M. McNett and G. M. Voelker, "Access and mobility of wireless pda users," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 2, pp. 40–55, 2005. Cited on page 31.

[59] Seongik, K. Hong, and Lee, "SLAW trace generator," http://research.csc.ncsu.edu/netsrv/?q=content/human-mobility-models-download-tlw-slaw, 2009, [Online]. Cited on page 33.

[60] I. Papapanagiotou, E. M. Nahum, and V. Pappas, "Smartphones vs. laptops: comparing web browsing behavior and the implications for caching," in *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, 2012, pp. 423–424. Cited on page 34.

[61] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "Youtube traffic characterization: a view from the edge," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, 2007, pp. 15–28. Cited on page 34.

[62] A. B. Downey, "The structural cause of file size distributions," in *9th IEEE Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2001, pp. 361–370. Cited on page 34.

[63] M. B. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær, "Entracked: energy-efficient robust position tracking for mobile devices," in *Proceedings of the 7th ACM International Conference on Mobile Aystems, Applications, and Services*, 2009, pp. 221–234. Cited on page 37.

[64] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using

mobile phones," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, 2009, pp. 85–98. Cited on page 37.

[65] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, 2010, pp. 85–98. Cited on page 37.

[66] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, 2008, pp. 323–336. Cited on page 37.

[67] C. V. Networking, "Global mobile data traffic forecast update, 2012'2017," *Cisco white paper*, 2013. Cited on page 40.

[68] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *Communications Magazine, IEEE*, vol. 44, no. 11, pp. 134–141, 2006. Cited on pages 40 and 69.

[69] M. Conti and M. Kumar, "Opportunities in opportunistic computing," *Computer*, vol. 43, no. 1, pp. 42–50, 2010. Cited on page 40.

[70] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008. Cited on pages 41 and 64.

[71] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010. Cited on pages 41, 47, and 64.

[72] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro, "Time-varying graphs and dynamic networks," in *Ad-hoc, Mobile, and Wireless Networks*. Springer, 2011, pp. 346–359. Cited on pages 43 and 44.

[73] M. Musolesi and C. Mascolo, “A community based mobility model for ad hoc network research,” in *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*. ACM, 2006, pp. 31–38. Cited on page 45.

[74] K. Nahrstedt and L. Vu, “CRAWDAD data set uiuc/uim (v. 2012-01-24),” Downloaded from http://crawdad.org/uiuc/uim/, Jan. 2012. Cited on page 46.

[75] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, “Pocket switched networks and human mobility in conference environments,” in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*. ACM, 2005, pp. 244–251. Cited on page 46.

[76] T. G. Griffin, F. B. Shepherd, and G. Wilfong, “The stable paths problem and interdomain routing,” *IEEE/ACM Transactions on Networking (ToN)*, vol. 10, no. 2, pp. 232–243, 2002. Cited on page 50.

[77] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, “Performance modeling of epidemic routing,” *Computer Networks*, vol. 51, no. 10, pp. 2867–2891, 2007. Cited on page 55.

[78] T. Camp, J. Boleng, and V. Davies, “A survey of mobility models for ad hoc network research,” *Wireless communications and mobile computing*, vol. 2, no. 5, pp. 483–502, 2002. Cited on page 64.

[79] M. Kim, D. Kotz, and S. Kim, “Extracting a mobility model from real user traces.” in *INFOCOM*, vol. 6, 2006, pp. 1–13. Cited on page 64.

[80] L. Pajevic and G. Karlsson, “A zero-dimensional mobility model for opportunistic networking,” in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*. IEEE, 2011, pp. 1–6. Cited on page 64.

[81] Y. Tseng, S. Kuo, H. Lee, and C. Huang, “Location tracking in a wireless sensor network by mobile agents and its data fusion strategies,” in *Information Processing in Sensor Networks*. Springer, 2003, pp. 554–554. Cited on page 66.

[82] R. Bajaj, S. Ranaweera, and D. Agrawal, "Gps: Location-tracking technology," *Computer*, vol. 35, no. 4, pp. 92–94, 2002. Cited on page 66.

[83] S. Qayyum, U. Sadiq, and M. Kumar, "Meme: Real-time mobility estimation for mobile environments," in *Local Computer Networks (LCN), 2014 IEEE 39th Conference on*. IEEE, 2014, pp. 133–141. Cited on page 66.

[84] S. C. Nelson, M. Bakht, and R. Kravets, "Encounter-based routing in dtns," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 846–854. Cited on page 67.

[85] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks." in *INFOCOM*, vol. 6, 2006, pp. 1–11. Cited on page 67.

[86] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 11, pp. 1576–1589, 2011. Cited on page 67.

[87] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2007, pp. 32–40. Cited on page 67.

[88] J. Ott, E. Hyytia, P. Lassila, T. Vaegs, and J. Kangasharju, "Floating content: Information sharing in urban areas," in *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*. IEEE, 2011, pp. 136–146. Cited on page 67.

[89] C. Boldrini, M. Conti, and A. Passarella, "Contentplace: social-aware data dissemination in opportunistic networks," in *Proceedings of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems*. ACM, 2008, pp. 203–210. Cited on page 67.

[90] S. Menard, *Applied logistic regression analysis*. Sage Publications, Incorporated, 2001, vol. 106. Cited on page 72.

[91] D. Hosmer and S. Lemeshow, *Applied logistic regression*. Wiley-Interscience, 2000, vol. 354. Cited on pages 73 and 74.

[92] Google, "Sensormanager," Dec 2014. [Online]. Available: http://developer.android.com/reference/android/hardware/SensorManager.html Cited on page 77.

[93] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong, "CRAWDAD trace," http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels/GPS/NC_State_Fair, July 2009. Cited on page 84.

[94] K. Lee, S. Hong, S. Kim, I. Rhee, and S. Chong, "Slaw: A new mobility model for human walks," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 855–863. Cited on page 85.