Contact Prediction, Routing and Fast Information Spreading in Social Networks

by

Kazem Jahanbakhsh
B.Sc., Sharif University of Technology, 2001
M.Sc., Sharif University of Technology, 2005

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Contact Prediction, Routing and Fast Information Spreading in Social Networks

by

Kazem Jahanbakhsh
B.Sc., Sharif University of Technology, 2001
M.Sc., Sharif University of Technology, 2005

Supervisory Committee

Dr. Valerie King, Co-Supervisor
(Department of Computer Science)

Dr. Gholamali C. Shoja, Co-Supervisor
(Department of Computer Science)

Dr. Kui Wu, Departmental Member
(Department of Computer Science)

Dr. T. Aaron Gulliver, Outside Member
(Department of Electrical & Computer Engineering)

Supervisory Committee

Dr. Valerie King, Co-Supervisor
(Department of Computer Science)

Dr. Gholamali C. Shoja, Co-Supervisor
(Department of Computer Science)

Dr. Kui Wu, Departmental Member
(Department of Computer Science)

Dr. T. Aaron Gulliver, Outside Member
(Department of Electrical & Computer Engineering)

ABSTRACT

The astronomical increase in the number of wireless devices such as smart phones in 21^{th} century has revolutionized the way people communicate with one another and share information. The new wireless technologies have also enabled researchers to collect real data about how people move and meet one another in different social settings. Understanding human mobility has many applications in different areas such as traffic planning in cities and public health studies of epidemic diseases. In this thesis, we study the fundamental properties of human contact graphs in order to characterize how people meet one another in different social environments. Understanding human contact patterns in return allows us to propose a cost-effective routing algorithm for spreading information in Delay Tolerant Networks. Furthermore, we propose several contact predictors to predict the unobserved parts of contact graphs when only partial observations are available. Our results show that we are able to infer hidden contacts of real contact traces by exploiting the underlying properties of contact graphs.

In the last few years, we have also witnessed an explosion in the number of people who use social media to share information with their friends. In the last part of this thesis, we study the running times of several information spreading algorithms in social networks in order to find the fastest strategy. Fast information spreading has an obvious application in advertising a product to a large number of people in a short amount of time. We prove that a fast information spreading algorithm should efficiently identify communication bottlenecks in order to speed up the running time. Finally, we show that sparsifying large social graphs by exploiting the edge-betweenness centrality measure can also speed up the information spreading rate.

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **v**

List of Tables **ix**

List of Figures **x**

Acknowledgements **xiii**

Dedication **xiv**

1 Introduction **1**

1.0.1 Main Contributions 3

1.0.2 Thesis Organization 4

2 Background and Related Work **6**

2.1 Related Work 6

2.1.1 Social Networks 6

2.1.2 Delay Tolerant Networks (DTNs) 9

2.1.3 Prediction in Social Networks 11

2.1.4 Information Spreading in Social Networks 12

3 A Socially-Based Greedy Routing Algorithm for Delay Tolerant Networks **14**

3.1 Socially-Based Greedy Routing 15

3.1.1 Real Data Description 15

3.1.2 Social Similarity/Distance Definitions 16

3.1.3 Human Network Mobility Model 17
3.1.4 Social-Greedy Routing Algorithms 17
3.2 Evaluation Methodology 18
3.2.1 Social-Sim Package 18
3.2.2 Results and Evaluations 20
3.3 Discussion 24

4 Human Contact Prediction using Contact Graph Inference 25
4.1 Problem Definition 26
4.2 Graph Inference using Social Information 26
4.2.1 Real Data Description 27
4.2.2 Jacard Social Similarity 27
4.2.3 Social Foci Similarity 27
4.2.4 Max Social Similarity 29
4.2.5 Graph Inference using Social Similarity 29
4.3 Graph Inference using Contact Graph Properties 32
4.3.1 Number of Common Neighbors (NCN) 32
4.3.2 Shortest Path (SP) 32
4.3.3 Random Walk (RW) 33
4.4 Contact Graph Model 35
4.5 Discussion 37

5 Predicting Missing Contacts in Mobile Social Networks 39
5.1 Problem Definition 41
5.2 Reconstructing Contact Graphs 41
5.2.1 Constructing Partial Contact Graphs 42
5.2.2 Contact Graph Properties 42
5.2.3 Methods Based on Neighborhood Similarity 43
5.2.4 Methods Based on Social Similarity 45
5.2.5 Method Based on Popularity 46
5.2.6 Reconstruction Algorithm 46
5.3 Performance Evaluation 47
5.3.1 Real Data Description 47
5.3.2 Testing Reconstruction Algorithm using Real Data 48
5.3.3 Why Predicting Missing Contacts? 49

5.3.4 Contact Prediction using Time-Spatial and Popularity Information . . . 51
5.3.5 Contact Prediction using Social Information . . . 56
5.3.6 Statistical Analysis of Contact Predictors . . . 58
5.3.7 NCN Scalability . . . 63
5.4 Why NCN Performs Well? . . . 65
5.5 Discussion . . . 67

6 Predicting Human Contacts using Supervised Learning 69
6.1 Related Work . . . 70
6.2 Problem Definition . . . 70
6.3 Contact Prediction using Classification Algorithms . . . 71
6.3.1 Logistic Regression Overview . . . 71
6.3.2 K-Nearest Neighbor Overview . . . 72
6.3.3 Features Extraction . . . 72
6.3.4 Training/Validating Binary Classifiers . . . 74
6.4 Prediction Results . . . 75
6.4.1 Approach I Results . . . 76
6.4.2 Approach II Results . . . 76
6.4.3 Features Significance . . . 77
6.4.4 Centrality Effect on Predictability . . . 78
6.5 Discussion . . . 80

7 Fast Information Spreading in Social Networks 81
7.1 Background and Related Work . . . 82
7.2 Problem Definition . . . 85
7.3 Empirical Analysis . . . 86
7.3.1 Real Data Description . . . 86
7.3.2 Empirical Analysis of Information Spreading Algorithms . . . 86
7.3.3 Finding Bridges as Communication Bottlenecks . . . 87
7.3.4 An Efficient Framework for Finding All k-cuts . . . 90
7.3.5 Discussion . . . 93
7.4 Mathematical Analysis . . . 93
7.4.1 The Effect of 1-whiskers on Information Spreading . . . 93
7.4.2 Hypothesis: A Core-Periphery Structure . . . 98

7.4.3 Discussion 99
7.5 Graph Sparsification 99
7.5.1 Sparsification using Betweenness Centrality 101
7.5.2 Discussion 101

8 Conclusions and Future Work **103**
8.1 Conclusions 103
8.2 Future Work 104

Bibliography **106**

List of Tables

Table 3.1 Correlation Coefficient 17
Table 3.2 Simulation Parameters 21

Table 4.1 Correlations 29

Table 5.1 Real Data Description 47
Table 5.2 Number of *k-cliques* (original/sampled contact graphs) 51
Table 5.3 The percentage of missing part of contact traces 51

Table 6.1 Approach I performance results (Logistic Regression/KNN) . . . 75
Table 6.2 Approach II performance results (Logistic Regression/KNN) . . 76
Table 6.3 The average rank for different features (Infocom 2006) 77
Table 6.4 Performance results of NCN linear classifier (Infocom 2006) . . . 78
Table 6.5 The effect of centrality on predictability 79

List of Figures

Figure 3.1 Successful Delivery Ratio for Different Routing Schemes (TTL=9h) 21
Figure 3.2 Total Delivery Cost for Different Routing Schemes (TTL=9h) 22
Figure 3.3 Successful Delivery Ratio for the three versions of Social-Greedy Algorithms (TTL=9h) 23
Figure 3.4 Total Delivery Cost for the three versions of Social-Greedy Algorithms (TTL=9h) 23

Figure 4.1 Nodes Belonging to Multiple Foci 28
Figure 4.2 Contact graphs with different threshold values 30
(a) Infocom 2005 30
(b) Infocom 2006 30
Figure 4.3 Performance of social profiles (Infocom06) 31
Figure 4.4 Performance of contact graph structure (Infocom05) 34
Figure 4.5 Performance of contact graph structure (Infocom06) 34
Figure 4.6 Performance of contact graph structure (synthetic mobility with $q = 1.0$) 36
Figure 4.7 Performance of contact graph structure (synthetic mobility with $q = 0.2$, $p = 0.2$, and $k = 5$) 36

Figure 5.1 Constructing the partial contact graph G_k 42
Figure 5.2 The average clustering coefficient (Infocom06: 9:00 AM to 6:00 PM) 43
Figure 5.3 The effect of common neighbors on geographical proximity . . 44
Figure 5.4 Simulating a partial contact graph 48
Figure 5.5 Contact duration distribution 50
Figure 5.6 Percentage of true positives for contact predictions (Infocom 2005) 52
Figure 5.7 Percentage of true positives for contact predictions (Infocom 2006) 53

Figure 5.8 Percentage of true positives for contact predictions (Cambridge) 53
Figure 5.9 Percentage of true positives for contact predictions (Roller) . . 54
Figure 5.10 Percentage of true positives for contact predictions (MIT) . . 54
Figure 5.11 Evolution of Contact Graph Densities for MIT and Infocom 2006 datasets 55
Figure 5.12 Percentage of true positives for contact predictions using social data (Infocom 2006) 56
Figure 5.13 Contact probability as a function of social and proximity information (Infocom 2006) 57
Figure 5.14 True positive rates of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06) 60
Figure 5.15 False positive rates of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06) 61
Figure 5.16 Precisions of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06) 61
Figure 5.17 Accuracies of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06) 62
Figure 5.18 RMSE of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06) 62
Figure 5.19 Scalability results for NCN predictor (Info06) 64
Figure 5.20 Scalability results for NCN predictor (Roller) 64
Figure 5.21 Number of predictions for NCN predictor (Infocom 2006/Roller) 65
Figure 5.22 Probability of contact as a function of *NCN* for Infocom06 dataset 66

Figure 6.1 Contact duration as a feature. 73
Figure 6.2 Class density distributions for lunch session (Infocom 2006) . . 78
Figure 6.3 Class density distribution of external nodes with highest centrality (Infocom 2006: keynote) 79
Figure 6.4 Class density distribution of external nodes with lowest centrality (Infocom 2006: keynote) 80

Figure 7.1 Running times of the random push-pull, Doerr, and Censor algorithms in the Facebook graph 87
Figure 7.2 A sample of detected 1-whiskers in Facebook graph 88
Figure 7.3 The number of 1-whiskers and 2-whiskers as a function of their sizes 88

Figure 7.4 Running times of random push-pull, Doerr, and Censor algorithms without 1-whiskers 89
Figure 7.5 Computing the maximal 3-edge connected component 91
Figure 7.6 The first 10 largest 2-whiskers and their corresponding nodes of the core 91
Figure 7.7 Components connectivity in a social network 92
Figure 7.8 Two types of 1-whiskers identified from empirical data 95
Figure 7.9 Well-connected core and weakly connected periphery 97
Figure 7.10 The edge-betweenness centrality distribution in Facebook . . . 100
Figure 7.11 Information spreading in Facebook by using the centrality sparsification technique 100

ACKNOWLEDGEMENTS

I would like to thank:

My Father, Sisters, and Yumi Moon for supporting me through my education.

Ali Shoja and Valerie King for mentoring, support, encouragement, and patience.

Uvic and NSERC, for fellowship awards and financial support.

Computer Science is no more about computers than astronomy is about telescopes.
Edsger W. Dijkstra

DEDICATION

I dedicate this Thesis to my Mother, may her soul rest in peace.

Chapter 1

Introduction

The appearance of new wireless technologies and smart phones has revolutionized the way people communicate and share information such as videos, photos, and messages. These new technologies have also allowed researchers to collect people's *contact* events by distributing a limited number of short range wireless sensors among them [22, 14, 78]. We say two people are in *contact*, if they are in close proximity of each other (e.g. Bluetooth range). The availability of contact traces has allowed researchers to identify the fundamental properties of human mobility and to propose realistic mobility models [66, 37].

By using these mobility models, researchers have proposed efficient routing protocols for Delay Tolerant Networks (DTNs) in which nodes exchange information when they are in close proximity of each others. In DTNs, the network is sparse and disconnected most of the time. Thus, most known protocols for Mobile Ad-Hoc Networks (MANETs) fail to operate in DTNs where successful delivery of a message strongly relies on human contact patterns. SimBet [19] and Bubble Rap [41] routing algorithms are a few examples in which nodes exploit the underlying properties of contact traces for optimal routing.

Proposing efficient routing algorithms for DTNs is a challenging task because human contacts are hard to predict. Several recent work showed the importance of *communities* (a community is set of nodes that are well-connected to one another; however, they are weakly connected to the rest of nodes in the graph) in efficient routing of messages in DTNs. However, real time community detection in DTNs is a complex and time consuming process. In this thesis, we propose a cost-effective method for bootstrapping wireless devices by exploiting available social profiles. In particular, we propose a greedy routing algorithm called *Social-Greedy* which uses

the social distance derived from people's *social profiles* to route messages to their destinations. Social profile of a user is the set of her social characteristics such as nationality, spoken language, affiliation and so on.

Predicting human mobility is complex because there are many parameters which influence the way people move. These parameters range from social factors such as people's occupations to the structure of the environment in which people move. Understanding the properties of human mobility has several applications in different areas. For example, it can be used to find the most efficient locations for GSM antennas. It can also be used for traffic planning in cities and public health studies of epidemic diseases. Researchers have studied different aspects of human behavior such as the way people become friends, call each others by their phones, or collaborate for publishing papers. They found the small-world network properties in most of these human-embedded networks [84].

In this thesis, for the first time, we formulate the problem of human contact prediction as a graph inference problem. We model human mobility by a *contact graph* in which nodes are people and edges are contact events between them. We show the importance of using offline social information for predicting people's contacts motivated by the homophily theory [61]. We also show that we can reconstruct hidden parts of contact graphs by using their small-world network properties when only parts of contact graphs are known. Our results are promising because they highlight the importance of using social profiles of people as well as the underlying structure of contact graphs for inferring the hidden parts of the graphs.

Experimentally measured contact traces, such as those obtained in a conference setting by using short range wireless sensors, are usually limited with respect to the practical number of sensors that can be deployed as well as the number of available human volunteers. Moreover, most previous experiments in this field can report only partial contact information since not everyone participating in the experiment carries a sensor device [22, 14, 78]. Previously collected contact traces have significantly contributed to the development of realistic human mobility models [66, 37] and efficient routing algorithms for DTNs where human contacts play a vital role in message delivery [19, 41, 48]. By exploiting time-spatial properties of contact graphs as well as popularity and social information of mobile nodes, we propose several novel methods to reconstruct missing parts of contact graphs where only a subset of nodes are able to sense contacts.

Next, we formulate the contact prediction problem in the context of machine

learning. In particular, we employ two well-known supervised classifiers for predicting hidden contacts among participants who carry cellphones. We extract several features by using information from contact graph structures, people's social profiles, and information of static sensors. The performance results of our supervised classifiers show the applicability of using machine learning algorithms for contact prediction tasks. Our results also show that a small subset of features such as the *number of common neighbors* and the *total overlap time* play essential roles in forming human contacts. Finally, we show that contacts of nodes with high centrality are more predictable than nodes with low centrality.

The appearance of online social networking services such as Facebook, Twitter, Flickr, Instogram, and many others has revolutionized the way in which information spreads in the world. Hudson river accident in 2009 [3] and Arab Spring in 2010 [1] are just a few examples of how fast information propagates in social media. Several information spreading algorithms have been proposed by researchers during the past few years [64, 12, 21]. However, the main challenge is to find out the algorithm with the lowest running time for spreading information in social networks.

We compare the running times of three well-known information spreading algorithms in the field by using the real data collected from the Facebook website [80]. We also mathematically prove the importance of the periphery of the Facebook social graph for the speed of information spreading algorithms. Our results highlight the effect of small communities that are weakly connected to the core of Facebook graph (i.e. *1-whiskers*) as the main communication bottlenecks for information spreading. Furthermore, we employ the *graph sparsification* technique to speed up information spreading in social networks. We exploit the *edge-betweenness centrality* measure [1] in order to identify communication bottlenecks and sparsify the graph by throwing out unimportant edges. Our results show that graph sparsification by using the edge-betweenness centrality efficiently spreads information in social networks.

1.0.1 Main Contributions

The main contributions of this thesis are threefold. First, different from other existing routing algorithms, we propose a cost-effective greedy routing algorithm which makes its routing decisions purely based on offline social profiles of people. By using a real human contact trace collected from a conference environment, we evaluate the per-

[1] The edge-betweenness centrality identifies edges that fall between clusters.

formance of our proposed socially-based greedy routing algorithm. Our performance results show that our routing algorithm is more cost-effective than Epidemic algorithm [79] and demonstrates a higher delivery ratio than Waiting routing algorithm [39].

Second, for the first time, we study the interesting problem of contact prediction by characterizing the most fundamental properties of human contact graphs for inferring hidden and future contacts. We devise a number of different unsupervised and supervised contact predictors by using offline social profiles of people as well as the underlying structure of human contact graphs. We evaluate the performance of our predictors by using several human contact traces collected from different social settings such as conferences, outdoor events, and campus environments [22, 14, 78, 54]. Our prediction results highlight the importance of the time-spatial properties of contact graphs and people's social profiles for the prediction task.

Third, we study the interesting problem of fast information spreading in social networks. We empirically and mathematically show that for fast information spreading in social networks one should efficiently identify and exploit communication bottlenecks. Finally, we empirically show that one can speed up information spreading in social networks by employing the edge-betweenness centrality definition in order to sparsify unimportant edges.

1.0.2 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, we review the related work on structure of social networks, DTNs, the prediction problem in social networks, and information spreading in social networks. The Social-Greedy algorithm description and its performance results are presented in Chapter 3. Chapter 4 describes our proposed methods for inferring hidden parts of contact graphs by using people's social profiles and the information extracted from underlying structure of contact graphs, and evaluates their prediction accuracies in details. In Chapter 5, we propose several prediction methods in order to complete the partial contact traces which were previously collected from different environments. Furthermore, in Chapter 6 we expand our results in Chapter 5 by employing two well-known supervised machine learning algorithms for contact prediction. In Chapter 7, we look into the problem of fast information spreading in social networks where the running time analysis of several information spreading algorithms as well as the importance of the

edge-betweenness centrality for information spreading are studied. Finally, Chapter 8 concludes the thesis and discusses future work.

Chapter 2

Background and Related Work

2.1 Related Work

In this chapter, we give an overview of the most important related work. First, we give a brief definition for social networks and discuss their main structural properties. Next, we look at DTNs and explain the most important properties which have been observed in human mobility in the last few years. Furthermore, we discuss the most influential routing algorithms for DTNs. The next area that we cover is the link prediction problem in social networks. We describe different approaches that researchers have taken to address this issue. Finally, we discuss the work that have been done recently in the area of information spreading.

2.1.1 Social Networks

A *social network* is a graph in which nodes are people and links between nodes can be any kind of relationships such as friendship. In social networks, if there is a link between two nodes, then we say those two nodes can communicate and send messages to each other even if they are geographically far away from one another. This communication can happen face to face, by a phone call, or by an email. In the last few years, the appearance and popularity of online social networking services such as Facebook and Twitter websites have opened a great opportunity for researchers to study properties of social networks as well as different aspects of human behaviour in online worlds.

Social Networks Properties

Several empirical experiments such as Milgram's show that social networks have low diameters [63]. This property can be modeled by random graphs. However, researchers have found that the underlying structure of social network is not completely random [84]. Instead, they have a high clustering coefficient which can only be found in regular graphs. This property is explained by "*Triadic Closure*" principle in that if u and v are two individuals with a common friend w, then there is a high probability for u and v to also become friends [23]. The Triadic Closure principle increases the local connectivity of the underlying graph which does not exist in random graphs. Watts and Strogatz addressed the two properties of having (1) low diameter, and (2) high clustering coefficient [84] in their small-world network model. In their paper, they argued that small-world networks fall somewhere between regular and random graphs. They constructed their small-world network by randomly adding long range edges between vertices placed on a circle.

Barabási and Albert noticed that many real world networks such as an actor collaboration graph and the World Wide Web have *power-law* [1] degree distributions [8]. A network whose nodes' degrees follow a power-law distribution is called a *scale-free* network. In scale-free networks, the exponent α of the power-law degree is typically between two and three [18]. Barabasi and Albert proposed the *Preferential Attachment* model in order to explain the appearance of power-law distribution in social networks.

Social Networks Navigability

In the early 1960s, the striking experiment, known as "six degrees of separation", was performed by Stanley Milgram and his co-workers [63]. In Milgram's experiment, 96 people from Massachusetts and Nebraska were asked to pass letters to their acquaintances in order to find an unknown distant person in Boston. They discovered that strangers in a social network are connected with a high probability through a short chain of friends. Milgram's experiment shows that not only do social networks have low diameters, but people can also successfully find these short paths by using only their local information. This feature of social networks is called efficient navigability since its expected delivery time is polylogarithmic in the size of network. In Milgram's

[1] The number of nodes with the degree d is proportional to $\frac{1}{d^{\alpha}}$ for a constant α.

experiment, at each step, node u, that holds a message, forwards the message to one of its neighbors v which is closest to the target node t in terms of social distance.

There exists an extensive literature on searching in social networks. In [52], Kleinberg proposed a small-world network model in order to explain the efficient navigability property in social networks. Watts et al. [83] also proposed a general framework to address the navigability property in social networks by presenting different social dimensions (e.g. geographical location, interest, and occupation) by a tree-based hierarchical structure. Two individuals are socially close if they are close in any social dimension. They performed a detailed simulation-based evaluation to study the effect of the number of dimensions and a homophily parameter on the performance of a searching strategy inspired by Milgram's experiment.

Adamic et al. proposed an efficient routing algorithm for peer-to-peer networks with power-law node degrees [6]. They found that by utilizing high degree nodes, the expected number of steps to deliver a message to its destination remains polynomial in the network size. As a result, although we cannot achieve an efficient routing only by utilizing nodes' degrees, taking advantage of high degree nodes can enhance the expected delivery time of the routing algorithm.

Dodds et al. repeated Milgram's experiment by emails [20]. Based on subject reports, they discovered that geographical proximity of acquaintance to the target dominates the early stages of a search while after the third step, similarity of occupation to the target is the dominant social dimension in passing a message to an acquaintance. Employing non-geographic dimensions for searching in social networks was the focus of several recent work. For example, Adamic et al. found that in an online student network where the acquaintanceship information is not complete and there is not a well defined hierarchical structure, local routing [2] strategies are not efficient [5].

Motivated by Kleinberg and Watts' work, Nowell et al. [59] studied the performance of a greedy routing strategy on a network structure crawled from LiveJournal. They investigated the effect of geographical distance on routing strategies when senders and receivers are geographically far from each other. Nowell et al. found that since the probability of being friend with a particular person is inversely proportional to the number of closer people in Livejournal, a greedy routing strategy can deliver messages efficiently.

[2]Their routing algorithm was a local routing since it was limited to people from the same school.

2.1.2 Delay Tolerant Networks (DTNs)

The appearance of new wireless technologies and the explosive increase in the number of people who carry cellphones have allowed researchers to study different aspects of human mobility. Meanwhile, in communication area we are experiencing a new wireless paradigms called DTNs which is different from traditional MANETs. In particular, DTNs or Mobile Opportunistic Networks are considered as those communication networks in that people carry their wireless devices such as cellphones and information can be potentially exchanged if two people are in close geographical proximity of each other (e.g. Bluetooth range).

In DTNs, routing a message to its destination strongly depends on how people contact each other over time. Therefore, understanding the fudamental properties of contact graphs play an essential role on the performance of any proposed routing algorithm for DTNs.

Human Mobility Properties

By using human mobility traces, researchers have found a close connection between people's social profiles and their mobility patterns. For example, Eagle et al. successfully inferred friendships from observational collected data from cell phones [67]. Their results demonstrate a distinctive connection between people's social profiles and their temporal and spatial mobility patterns. Moreover, Mtibaa et al. showed that there is a strong correlation among properties of nodes, links, and paths in social and contact graphs [65].

Researchers proposed several synthetic mobility models based on underlying properties of contact graphs. Miklas et al. proposed a mobility model for friends and strangers networks by using Watts and Strogatz's small-world network and Preferential Attachment models, respectively [62]. Musolesi et al. proposed a community-based mobility model (CMM) in which nodes tend to contact other nodes from their own community with higher probability than nodes from different communities [66]. They validated CMM by testing to see if it produces the same distributions for inter-contact[3] and contact duration times as the ones that were observed in real contact traces. By analyzing human contact and wireless LAN traces, Hsu et al. also introduced the time-variant community model for human mobility [37]. They observed a

[3]Inter-contact time is the time gap separating two consecutive contacts between the same pair of nodes.

skewed probability distribution for locations where people visit every day as well as a time periodic pattern in human mobility. Most recently, researchers have used short range wireless sensors to collect contact data among soccer players [75]. They proposed a mobility model for soccer games based on the observed statistical properties from their contact traces.

By using human mobility traces, researchers have found that people 's popularity is directly related to the frequency with which they meet other people. Herrmann employed the Preferential Attachment model to satisfy this property in their human mobility model [35]. Hui et al. [38] employed the *betweenness centrality* metric (i.e. the number of times a node appears in shortest paths between all possible pairs of nodes in the network) similar to Freeman's betweenness centrality [26] to find central mobile nodes. They ran a large number of simulations of unlimited flooding with different uniformly distributed traffic profiles on human mobility traces. By counting the number of times a node acts as a relay for other nodes on all the shortest delay paths, they could measure the centrality of each node in contact graphs.

Routing in Delay Tolerant Networks (DTNs)

There has been much work done on routing in DTNs. However, we just focus on the most significant ones here. Vahdat et al. proposed the epidemic routing [79] which is similar to the blind flooding routing. Lindgern et al. also introduced a probabilistic routing called PROPHET [60] for DTNs which takes advantage of community structure in contact graphs to make routing decisions. By employing a priori affiliation information, Hui et al. proposed a routing algorithm called LABEL which takes advantage of communities for routing messages [39]. LABEL partitions nodes into communities only based on affiliation information which is collected from offline questionnaire forms.

Hui et al. also observed a variation in nodes' centralities in human contact traces [40]. They proposed the Bubble Rap routing algorithm which utilizes nodes communities and centralities for making routing decisions. The Bubble Rap uses nodes' centralities to reach the community of destination node quickly [41]. When a message reaches the community of destination node, Bubble Rap limits the message forwarding range to nodes with the same community as destination.

Similar to Bubble Rap, SimBet routing also exploits community structure and hetregenous nodes degree centralities of contact graphs for routing messages to their

destinations [19]. Hossmann et al. improved the performance of SimBet and Bubble Rap algorithms by proposing a strategy for mapping contact graphs to social graphs. Using their strategy, they showed that nodes can make their routing decisions more efficiently [36] than SimBet and Bubble Rap.

Miklas et al. studied the MIT's Reality Mining data [67], and demonstrated the importance of using social information in designing routing protocol, firewall for slowing down spreading of worms, as well as a P2P file sharing for opportunistic networks [62]. Boldrini et al. also proposed a context-aware routing protocol for opportunistic networks where every node captures context information of its neighbours and nodes which it has met in the past [10]. Based upon the collected social information, every node calculates the delivery probabilities of encountered nodes to choose the best candidate for routing messages.

2.1.3 Prediction in Social Networks

Given a snapshot of a social network, it is important to see if we can predict future new links or hidden links between people. In particular, suppose there is not any link between two nodes such as u and v at time t, it is interesting to formulate the probability that edge (u, v) appears at time $t + 1$. This problem is commonly known as the "*Link Prediction*" problem. There are a large number of work in which researchers have addressed the link prediction problem by taking data mining and machine learning approaches.

One of the early works in this area was done by Nowell et al. who studied the link prediction problem in a citation network [57]. They devised several predictors by extracting different graph topological features. They showed that methods based on the ensemble of all paths between two nodes in a social graph as well as the number of common neighbors method significantly perform better than others. Their work motivated us to extract several topological features from contact graphs for predicting missing contacts [49].

Goldberg et al. assessed the confidence of experimentally observed interactions among proteins by using cohesive neighborhoods and small average distance between proteins [27]. Their work motivated us to propose three prediction methods based on the small-world network properties of contact graphs in order to predict unobserved contacts [47] Authors of [50] introduced the Triad Transition Matrix (TTM) to compute the transition probabilities of 64 possible Triads for dynamic email-based

networks. They showed that TTM of different social networks have similar pattern. Finally, they employed the computed transition probabilities from TTMs for predicting future links in an email-based network.

Song et al. studied the limits of predictability in human mobility by studying the mobility patterns of cellphone users [76]. They discovered a high degree of regularity in human mobility resulting in a potential 93% predictability in user mobility. Vu et al. exploited the high level of regularity in human mobility in order to predict locations where a person will go and people she will contact in a given time of a day [81]. Wang et al. used mobile phone data and found a strong correlation between individuals' movements and their connectedness in their social network [82]. They devised both unsupervised and supervised predictors for inferring the new links in social networks by using information from people's social networks and their mobility patterns.

2.1.4 Information Spreading in Social Networks

One of the most interesting problem in the area of social computing is studying the process by which information propagates through social networks. Diffusion of an idea or opinion is a socio-economic problem and it has many things in common with our problem in Chapter 7 which is fast information spreading in social networks. Viral marketing is the area in which economists focus on finding efficient ways to advertise a new product to a large number of people quickly. Spreading an idea through a social network was studied thoroughly by Everett Rogers in his famous book "Diffusion of Innovations" [72]. Rogers defined the diffusion as the process by which a new idea is communicated through certain channels, e.g. word-of-mouth, among social network members over time. He also identified four basic elements that are crucial for diffusion process: an innovation, communication channels, time, and a social network.

According to the theory of diffusion of innovations, first of all, an innovation is more likely to be adopted by people who are seeking new ideas (i.e. *innovators*). If innovators like an idea or product, they tend to adopt it, and they are likely to share the new idea with their friends through communication channels. Second, another group of people called *early adopters* are strongly connected to innovators in the social network and follow innovators' opinions. The new opinion then spreads to other less influential groups of people who are *early majority*, *late majority*, and *laggards*, respectively. Rogers and other researchers have discovered a strong relationship

between diffusion process and social network structure [72].

There are a large number of works in social networking area in which researchers focused on understanding how information spreads in social networks through social influence. Kemp et al. aimed to find a subset of k influential nodes such that if one targets them as innovators, they would spread the message to the maximum number of nodes in the network [51]. For modelling information diffusion, they employed two different probabilistic processes. They compared their proposed greedy algorithm results for choosing the influential nodes with other strategies such as choosing the nodes with the highest degree or closeness centralities as the influential nodes. Their results highlight the importance of nodes' positions in a social network for being socially influential.

Authors of [13] studied the effect of social links on spreading popular photos in the Flickr website. They found that the social links between Flickr members play an essential role in information propagation in Flickr. They also discovered that the peer pressure plays an important role in making a photo popular. In other words, they showed that people are more likely to become fan of a photo as the number of their friends who already marked that photo as favorite increases. They emphasized that using a push-based strategy for spreading popular photos may work faster than the Flickr's current pull strategy. The authors of [58] studied the spreading of Internet chain letters and discovered that chain letters proceed in a narrow but very deep tree-like pattern. Most of our observations for diffusion of information in web or blogs are about when different nodes mention a piece of information or adopt it. However, the underlying diffusion network through which information spreads is unobservable. In [29], authors proposed an algorithm to infer the diffusion network given adoption times for nodes.

Finally, Goldenberg et al. analyzed the effect of word-of-mouth on marketing new products [28]. They studied the importance of three parameters for modelling diffusion of a new product including advertisement, strong ties, and weak ties. While strong ties model the strong social relationship between a person and her close friends from the same personal group, weak ties model the influence of people on their acquaintances and colleagues who belong to different personal groups. They did simulation by using Cellular Automata, and studied the speed of information dissemination and the effect of each of the above three parameters on the speed of influence. They found that weak ties are as important as strong ties to spread information in social networks.

Chapter 3

A Socially-Based Greedy Routing Algorithm for Delay Tolerant Networks

The appearance of smart phones has created a new opening for pervasive computing. People who carry these devices form a DTN in that senders can opportunistically forward messages to other mobile nodes in order to reach the destinations of the messages. These opportunistic wireless networks have two dimensions. One relates to the properties of wireless networks while the second dimension has a social network component based on human mobility pattern and people's social profiles. Unlike the legacy MANETs for which the existence of an end-to-end path between source and destination nodes is assumed, opportunistic networks have a more dynamic nature. Thus, routing in mobile opportunistic networks becomes a challenging task.

Milgram's experiment [63] shows that people can use their local information to successfully find short paths to destinations in which a message holder forwards its message to one of its neighbors which is in the closest social distance to the destination of the message. Kleinberg used a d-dimensional lattice as the underlying structure for his small-world network to model routing in social networks [52]. He employed a greedy routing algorithm similar to Milgram's to route messages to their destinations. Kleinberg showed that there does exist an efficient routing algorithm if there is a correlation between the underlying lattice structure and the length of long range connections. Our Social-Greedy routing algorithm proposed in this chapter is inspired by Miligram's experiment and Kleinberg's model.

Recent work have shown the importance of communities for routing information in DTNs [39, 41]. Hui et al. proposed three distributed algorithms for community detection in DTNs [42]. The high cost of information exchange and calculations, and the complexity of adjusting several threshold parameters required by distributed community detection algorithms have led us to consider static programming of mobile devices with pre-existing social profiles instead of running a dynamic method for detecting communities in contact graphs [1]. In this chapter, we assume that the social profiles of all participants are known in advance. The results of this chapter was published as a paper in Mobile Opportunistic Networking workshop in 2010 [48]. Our contributions in this chapter may be summarized as follows:

- Based on our results, we suggest a new low cost and simple method for bootstrapping mobile wireless devices with already available social information.
- Using social profiles collected from Infocom 2006's participants, we define a social distance and introduce three greedy routing algorithms for DTNs which are more cost effective than Epidemic routing [79] and have higher delivery ratio than Waiting routing [41].
- To our knowledge, this is the first empirical evaluation of Kleinberg's greedy strategy for a mobile network, and the first empirical evaluation for a routing strategy which uses social distance rather than geographic distance to determine each move.

3.1 Socially-Based Greedy Routing

In this section, we first propose a social distance by which we try to model human mobility in a conference environment. Next, we describe the three variations of our Social-Greedy algorithms. Finally, we compare the performance of our Social-Greedy algorithms with other existing routing algorithms in the field.

3.1.1 Real Data Description

Here we make use of the human mobility traces collected from a conferences environment. The dataset contains mobility traces of 79 researchers attending the Infocom

[1] A contact graph is a dynamic graph in which nodes are people and the edges between nodes are the encounters from human mobility traces.

2006 conference [74]. The experiment lasted for three days in which contacts between participants were recorded by using iMote sensors. These Bluetooth sensors sampled a contact between two people when they were in close proximity of each other (e.g. < 10 meters). In Infocom 2006 dataset, social profiles of people who participated in the experiment were collected. These social profiles included information about participants' (1) *nationality*, (2) *spoken languages*, (3) *current affiliations*, (4) *city and country of residence*, (5) *graduate school*, and (6) *research interests*.

3.1.2 Social Similarity/Distance Definitions

Probably the most natural way to compute the similarity between social profiles of two people is to use the Jacard index [43]. As mentioned earlier, our social profiles contain information about six different social dimensions. We denote each social dimension of every node as a set of features. For example, suppose node u speaks English and Spanish while node v speaks English and French. Let us denote English, Spanish, and French languages with numbers 1, 2, and 3 respectively. We show the spoken languages of nodes u and v with two sets $\Gamma_u^2 = \{1, 2\}$ and $\Gamma_v^2 = \{1, 3\}$ independently. By using Jacard index, we define the similarity between two nodes with respect to the social dimension i as follows [48]:

$$\sigma_{jacard}^{i}(u, v) = \frac{|\Gamma_u^i \bigcap \Gamma_v^i|}{|\Gamma_u^i \bigcup \Gamma_v^i|}, \tag{3.1}$$

where Γ_u^i is the feature set of node u for social dimension i and $|\Gamma_u^i|$ is its cardinality. The σ_{jacard}^{i} is a real number in $[0, 1]$. Assuming that we have d different social dimensions for each node, we define the total social similarity between two nodes by computing the total average over all d dimensions as follows:

$$sim_{jac}(u, v) = \sum_{i=1}^{d} \frac{\sigma_{jacard}^{i}(u, v)}{d}, \tag{3.2}$$

where $sim_{jac}(u, v)$ is the total social similarity between two nodes u and v by using the Jacard index and d is the number of social dimensions (for Infocom 2006 data: $d = 6$). Here, we assign the same weight to different social dimensions. Finally, we define the social distance between two nodes u and v as follows:

$$dist(u, v) = \begin{cases} sim_{jac}(u, v)^{-1} & \text{if } sim_{jac}(u, v) \neq 0 \\ \infty & \text{otherwise} \end{cases} \tag{3.3}$$

Table 3.1: Correlation Coefficient

Variable Pairs	Correlation
$r_{nc_{u,v}cd_{u,v}}$	0.369
$r_{cdu,vsdu,v}$	0.253
$r_{ncu,vsdu,v}$	0.182

where we say that u is socially closer to v than w if $dist(u,v) < dist(u,w)$. The reader should note that the symbol ∞ represents a very large number for the distance between two completely dissimilar nodes.

3.1.3 Human Network Mobility Model

Our main hypothesis is that in a conference environment people who are interested in the same research area or speak the same language have a higher probability to meet each other for a longer time than others. Therefore, we expect that for a given node u, the probability of meeting other nodes is influenced by u's social distance from other nodes. We find a close relationship between number of contacts and contact duration in the human contact trace collected in Infocom 2006 which is in agreement with previous work [40]. We also calculate the correlation coefficients between the number of contacts, the average contact duration time, and the social distance as shown in Table 3.1.

3.1.4 Social-Greedy Routing Algorithms

In the previous section we found a distinctive relationship between social distances among people and their mobility patterns. This motivates us to employ a greedy mechanism similar to Milgram's to route messages to their destinations. We assume that every node has information about the social profile of the destination, and it can also exchange its social profiles with any contacted node. We implement three versions for Social-Greedy routing as listed below:

- Social-Greedy I: If node u has a message for the destination v and encounters node w which is socially closer to v than u, u hands off that message to w. Node u does not remove the message from its buffer unless it encounters node v or the TTL of the message expires.

- Social-Greedy II: When node u, which is carrying message M for node v, encounters node w at time t_0, u hands off M to w if it is socially closer to v than u. However, for any $t > t_0$, u can only pass the message M to an encountered node if it is socially closer to v than w (the closest node to v which u has met so far).

- Social-Greedy III: When node u hands off the message M to w as in the first version, it deletes M from its own buffer.

While Social-Greedy I hands off the message M to any encountered node which is socially closer to destination of M, Social-Greedy II acts more conservatively in the sense that at each step, it limits the range of nodes which can be recipient of the message M based on the closest node to the destination that u has met so far. Furthermore, Social-Greedy III tries to efficiently utilize nodes' buffers by removing the message M from the buffer of the message holder after forwarding M to the next node which is socially closer to the destination. Therefore, we expect that Social-Greedy I to have the best success delivery ratio (SDR) [2] while Social-Greedy III to have the lowest total delivery cost.

3.2 Evaluation Methodology

In this section, we evaluate the performance of our Social-Greedy algorithms and compare it with other existing routing algorithms.

3.2.1 Social-Sim Package

We have developed the Social-Sim package in C++ [2] in order to analyze important properties of human mobility patterns in different social settings such as conferences, campus environments, and outdoor events. This package has the implementation for reading human mobility traces collected by Bluetooth sensors such as the MIT or Cambridge datasets [22, 54] and storing them in mysql database for post-processing steps. All contact traces contain contact information recorded by wireless sensors. A *contact event* between two wireless sensors shows that their owners were in close physical proximity of each other. A contact event can be represented by a quadruple

[2] SDR is the proportion of messages that have been successfully delivered out of the total unique messages.

(id_1, id_2, t_s, t_e) where id_i, t_s, and t_e represent the node id, start time and end time of the contact event, respectively.

The Social-Sim package has also the implementation for a discrete event simulator for comparing the performance of our Social-Greedy routing algorithms with other routing algorithms proposed for Delay Tolerant Networks (DTNs). During the execution of a routing algorithm, the Social-Sim records all important information including the number of delivered messages, their delivery delay, and the number of transmissions for each message. Using the *contact graph* definition in which nodes represent wireless sensors and edges represent contact events between sensors, the Social-Sim package allows researchers to analyze important statistical properties of contact graphs and to understand how people contact each others in different social settings. The Social-Sim package contains the following important features:

- Parser class: this class includes all necessary methods for reading the recorded data by wireless sensors which are in the text formats and parsing them into a mysql database for post-processing steps.

- Message class: this class basically implements a message entity that represents a piece of information such as a video, a text message, or a photo. Every message has a unique id as well as a sender and a receiver id. A message can carry extra information. When two mobile nodes come to the close proximity of each other, they can exchange their messages on demand.

- Traffic class: it implements the traffic flow concept by which every node can have some information which should be sent to another node in the network. Thus, in the beginning of the analysis we generate the traffic profiles for all nodes. Once a message is delivered to its destination, we update the traffic profile list to keep track of the delivered messages, their delivery costs and delays, and the number of hops taken by each message to reach its destination.

- Routing class: this class implements the three variations of the Social-Greedy routing algorithms as well as other routing algorithms proposed by other researchers for DTNs including Waiting [41], Epidemic [79], Label [39], Bubble rap [41], and so on.

- Contact Graph class: it has the required methods to generate a contact graph by using the real data collected by wireless sensors. It also contains several methods

for computing the centrality of nodes, the similarity between consecutive contact graphs over time, average clustering coefficient of a contact graph, and the neighborhood similarity between two nodes in a contact graph.

- Distribution class: this class has several methods to compute the probability distributions of different properties of contact graphs such as distribution of contact duration, inter-contact duration, and number of contacts between mobile nodes. It can also process social profiles of people in order to compute social distances between them. It implements the Jacard index [43] and the Social Focus [53] for computing the social distances between nodes. Furthermore, it calculates the rank of each node by counting the number of visited nodes. Finally, it has another method for finding the places where a person has visited during a time period.

- Statistics class: this class computes the mean and standard deviation of a data vector, as well as the Pearson correlation coefficient between two data vectors.

- RGModel class: this class has two methods for computing the Jacard similarity and the size of Social Focus between two nodes with respect to a social dimension such as research interest.

- main(): this method demos how to use different classes of the Social-Sim package.

3.2.2 Results and Evaluations

The simulation parameters are shown in Table 3.2. For our simulation, we use the contact data collected from the first day of the Infocom 2006 conference from 9:00 AM to 6:00 PM where 79 people participated. We assume that there are 1000 unique videos randomly distributed among all 79 nodes. The destination of each video is randomly selected among all nodes. We repeat our simulations 20 times and the presented results show the average values.

For evaluation and comparison purposes, we test a number of different routing algorithms including Epidemic [79], Waiting [41], LABEL [39], Bubble rap [41], and three versions of our Social-Greedy algorithms. In Waiting algorithm, a node that is carrying a message has to wait until it has a direct contact with the destination. In Epidemic, a message is given to any node that comes within the proximity of the

Table 3.2: Simulation Parameters

Parameter	**Value**
No of nodes	79
No of unique videos	1000
No of runs	20
Starting time	Apr. 24, 9:00 AM
TTL	9 hours
Communication type	Bluetooth 2.0

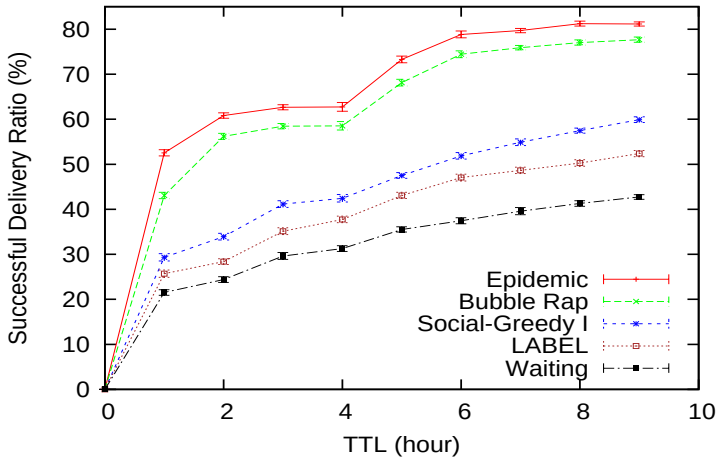


Figure 3.1: Successful Delivery Ratio for Different Routing Schemes (TTL=9h)

message holder provided that the receiver does not already have the message. Hence, Epidemic has the lowest bound for delivery delay and Waiting has the lowest bound for delivery cost. While LABEL passes a message to those nodes which have the same affiliation as the destination, Bubble rap uses nodes' affiliations and centralities for routing. We approximate nodes' centralities by measuring the number of unique nodes every node has met per hour [41]. We update the measured centralities every ten minutes to adapt to the dynamic of the environment.

We choose successful exchange of video files between nodes as a measure of their contact duration. The average video lengths are chosen to be 8.4 MBytes as it was observed from crawling Youtube website [15]. We also assume that all wireless devices have Bluetooth 2.0 which supports 3Mbit/sec data rate. Therefore, the average transmission time is calculated as $T_{trans} = \frac{8.4 \times 8}{3} \approx 23s$. This implies that minimizing the number of video transmissions per contact is important for a video sharing application.

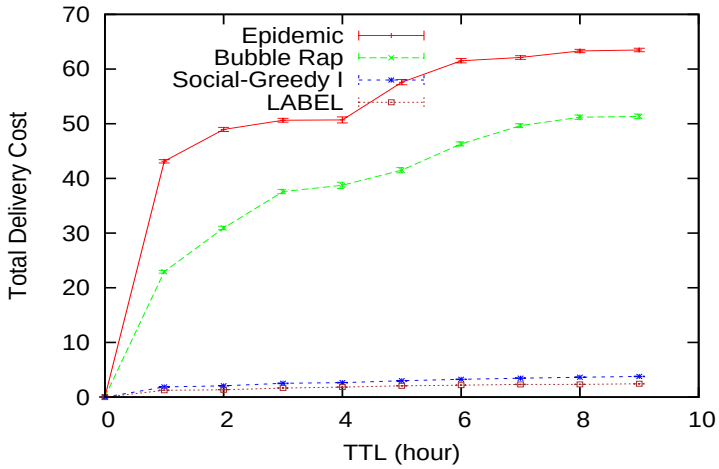


Figure 3.2: Total Delivery Cost for Different Routing Schemes (TTL=9h)

Figure 3.1 compares SDRs of different routing algorithms. It shows that Epidemic delivers around 81% of videos while Bubble rap and Social-Greedy I deliver 70% and 60%, respectively during the first 9 hours. According to Figure 3.1, Social-Greedy I in the worst case betters the SDR of LABEL and Waiting algorithms by 15% and 40%, respectively.

Furthermore, we compare the total delivery cost [3] of different routing algorithms. Figure 3.2 shows that Social-Greedy I forwards video files 17 times less than Epidemic and around 4.5 times less than Bubble rap. The low cost of Social-Greedy I algorithm shows that it can be considered as a power-efficient protocol for low-powered mobile devices.

To show the power of social profiling, we compare Social-Greedy routing with a Random routing. For fair comparison, first we use the same simulation parameters for both strategies. Second, in Random routing, we assign social distance in random and employ a hand off probability, which is the probability of forwarding on each encounter, to guarantee the same average costs for both strategies. Interestingly, Figures 3.3 and 3.4 show that both Social-Greedy I and II outperform LABEL and Random in terms of SDR while Social-Greedy II and III have lower costs than LABEL.

Our results are quite impressive considering the simplicity of using already available online social profiles from conference websites or any online citation network for making forwarding decisions. For example, we can easily download the list of all

[3]Total delivery cost is the total number of messages (including duplications) transmitted across the network. We normalize the total delivery cost by dividing it by the total number of unique messages.

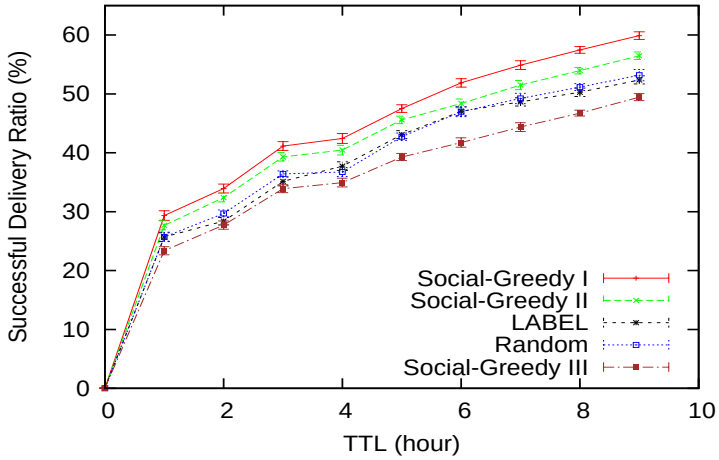


Figure 3.3: Successful Delivery Ratio for the three versions of Social-Greedy Algorithms (TTL=9h)

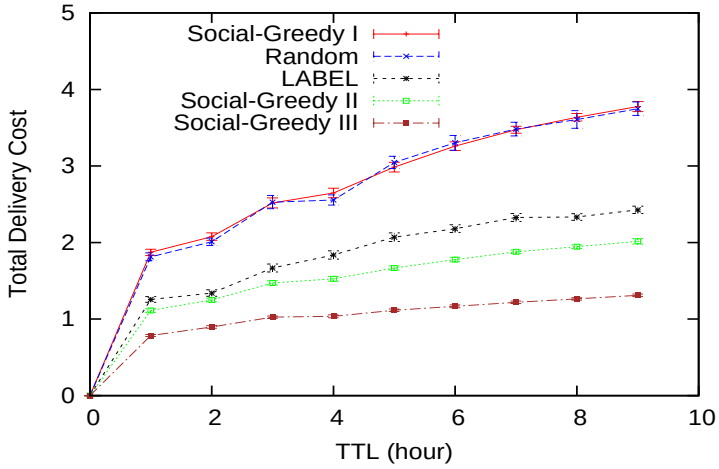


Figure 3.4: Total Delivery Cost for the three versions of Social-Greedy Algorithms (TTL=9h)

submitted papers from the Infocom 2006 website including the titles of papers and authors' names, schools, and city/country of residence. The titles of papers reflect the authors' topics of interests while the nationality and spoken language can be simply inferred from authors' names and their place of residence. Therefore, we are able to collect a rich social profile for each conference participant and use the collected information to initialize and bootstrap the mobile devices of the participants of the conference in advance.

3.3 Discussion

In this chapter, we employed the social profiles collected from questionnaire forms completed by Infocom 2006 conference attendees to enhance routing in mobile opportunistic networks. We used the Jacard measure of social profiles to define a social distance. Using defined social distance, we proposed a greedy routing algorithm inspired by Kleinberg's model. Moreover, we showed the effectiveness of using various social dimensions particularly for media sharing applications. Finally, we proposed a simple method for programming mobile nodes by using available social profiles.

Chapter 4

Human Contact Prediction using Contact Graph Inference

Everybody in a society is identified by a set of social characteristics such as occupation, affiliation, place of living, and so on. We call a person's set of social characteristics her social profile. The homophily phenomenon in which similar people are more likely to interact with each others has been studied in social networks [61]. Similarly, we believe that people are more likely to interact with those who are socially similar to them. In the previous chapter, we studied the effect of social characteristics on the people's mobility pattern in a conference setting.

In this chapter, we focus on predicting human contacts in a conference environment. We say two people are *in contact* if they are in close proximity of each others. For the first part of the chapter, motivated by homophily theory we investigate the importance of social similarity on people's mobility patterns [61]. We try to infer people's contacts by computing the similarities between their social profiles. In this part we assume that we only have information about people's social profiles while people's contacts are completely unknown. In the second part of the chapter, we study the same problem but in a different setting where we try to infer the missing contacts among people when only part of the contact graph, which is the graph constructed by nodes' mobility, is known. Inferring missing parts of a contact graph is important because it allows researchers to reconstruct unobserved part of contact graphs when there is a partial observation for people's contacts. The results of this chapter was published as a paper in Social Computing and Networking conference in 2010 [47]. The main contributions of this chapter are:

- We show the importance of social profiles as well as the underlying structure of contact graphs in contact prediction problem.
- We present several methods to reconstruct a contact graph when we only have information about people's social profiles or when only a partial part of a contact graph is known.

4.1 Problem Definition

In this chapter, our main problem is graph inference when a prior knowledge about the graph is available. In the first part of the chapter, for a graph $G = (V, E)$ we assume that we have offline information (e.g. social information) about vertices $v \in V$ while the edges in $E \subset V \times V$ are totally unknown. Thus, the problem becomes inferring edges in E by using the available side information about vertices in V. In the second part of the chapter, we assume that for a graph $G = (V, E)$ our vertices set is $V = V_{int} \cup V_{ext}$ where V_{int} and V_{ext} denote the internal and external vertices respectively. We also assume that all edges in $E_{known} \subset V_{int} \times (V_{int} \cup V_{ext})$ are known whereas all edges in $E_{unknown} \subset V_{ext} \times V_{ext}$ are missing ($E = E_{known} \cup E_{unknown}$). For such a partial graph, our problem becomes to infer the edges among external vertices (edges in $E_{unknown}$).

4.2 Graph Inference using Social Information

To model social interactions between people in a conference, we use a weighted contact graph $G = (V, E)$ where V is the set of people who attend in the social meeting and E is the set of edges between them. There is an edge $e = (u, v)$ in G between u and v if they contacted each other at least once. In the contact graph G, we assign a weight to the edge (u, v) which shows either the total number of times that u and v saw each other or the total time period that they spent together during the social event. This weight represents the strength of social relation between the corresponding nodes. We also assume that contact graph G is undirected because the social interaction involves both sides. For the first part of our analysis, we assume that the set of edges in E is unknown while there is social information for nodes in V.

4.2.1 Real Data Description

In this chapter, we use the human mobility traces collected from two different conferences [74]. The first dataset is collected during the Infocom 2005 conference where 41 participants of the conference attended the experiment. The second dataset contains mobility traces of 79 researchers attending the Infocom 2006 conference. The reader can find the details of the Infocom 2006 dataset in Chapter 3. Both experiments lasted for three days where contact events between participants were recorded by using iMote sensors. These Bluetooth sensors sampled a contact event between two people when they were in close proximity of each other. For the Infocom 2005 data, there is not any social information about participants. However, in the Infocom 2006, social profiles of people who participated in the experiment were also collected.

Based on the homophily theory, people who are socially close are more likely to be friends. The hypothesis that we want to test can be stated as below:

> *Hypothesis: (Social proximity) Individuals who have similar social profiles are more likely to contact each other than those who do not have similar social profiles.*

To test this hypothesis, we should first define the social similarity between nodes.

4.2.2 Jacard Social Similarity

As it was mentioned in the previous chapter, the Infocom 2006 dataset contains information about six different social dimensions of participants. One way to compute the total social similarity between two nodes is by using Equation 3.2 which is based on the Jacard index [43].

4.2.3 Social Foci Similarity

Let us define the *social focus* as a set of people who share the same research interest, speak the same language, or were born in the same country. Foci are a way of summarizing many possible reasons that two people contact each other: because they are from the same country, have the same affiliation, or share the same interests. Now, suppose in a conference there are two people u and v who are interested in the *Routing* research area. Moreover, assume that these two people do not have any other similarities with respect to other social dimensions. More specifically, they have

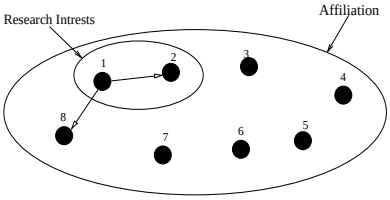


Figure 4.1: Nodes Belonging to Multiple Foci

different affiliations, were born in different countries, speak in different languages and so on. Roughly speaking, there is a high probability for these two people to meet each other in the conference because both of them may attend the same sessions. We assume that their contact probability has an inverse relationship with the number of people who are interested in the *Routing* area.

We define the *Foci distance* between two given nodes as the cardinality of the smallest social focus that both of them belong to. In the previous example, the social distance between u and v with respect to research interest is equal to the number of conference participants who are also interested in *Routing*. We write the Foci distance between two given nodes u and v as follows [53]:

$$d_{foc}(u, v) = min\left|\{F|u, v \in F\}\right|, \tag{4.1}$$

where F is the social focus that both u and v belong to. Note that there is a super group which contains all nodes. Considering Equation 4.1, we define the *Foci similarity* between two nodes as follows:

$$sim^{soc}_{foc}(u, v) = \frac{1}{d_{foc}(u, v)} \tag{4.2}$$

To highlight the main difference between the social Foci and the Jacard similarity, we show a set of 8 nodes in Figure 4.1. Suppose all of these 8 nodes have the same affiliation, nationality, school, language, and city. Furthermore, suppose all nodes share the same research interests except 1 and 2 which have a similar set of interests that is different from the rest of nodes (nodes 3 to 8). By using the Jacard index, we can show that $sim_{jac}(1, 2) = 1.0$ and $sim_{jac}(1, 3) = 0.83$. Thus, based on the Jacard index node 1 is almost at the same distance from both nodes 2 and 3. However, the Foci distance gives us $sim_{foc}(1, 2) = 0.5$ and $sim_{foc}(1, 3) = 0.125$. As we can see the Foci distance shows a closer distance between $(1, 2)$ than $(1, 3)$ because both nodes 1 and 2 share the same interests. Therefore, the Foci distance can separate those nodes

Table 4.1: Correlations

$\rho_{nc,jac}$	$\rho_{cd,jac}$	$\rho_{nc,foc}$	$\rho_{cd,foc}$
0.10	0.30	0.17	0.32

which are socially close from other nodes more significantly.

4.2.4 Max Social Similarity

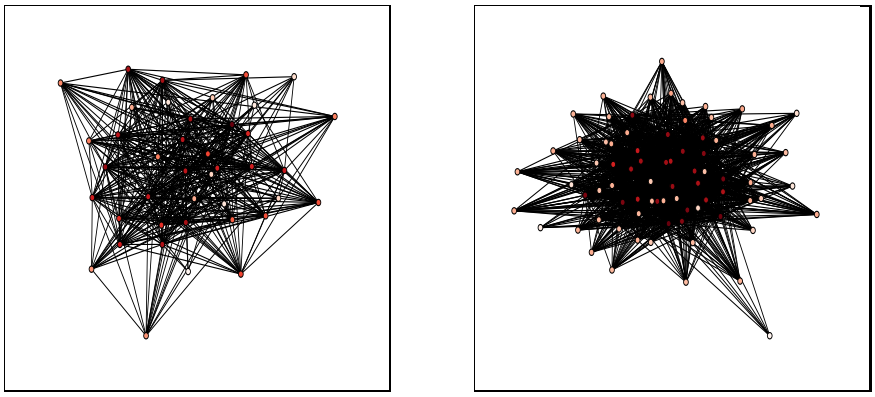
Watts et al. used a set of social characteristics to identify nodes in a social network [83]. They have defined the social distance between two nodes as the minimum distance over all dimensions. Combining their social distance with Equation 3.1, we introduce a new social similarity as follows:

$$sim_{max}(u,v) = \max_i \sigma^i_{jacard}(u,v) \tag{4.3}$$

Here, we assume all dimensions have the same weights, and if two nodes are similar in any dimension, they are assumed to be socially close to each other. As the first step, we calculate the Pearson correlation coefficient between social similarity of all pairs of nodes and their total number of contacts, and total contact durations for the Infocom 2006. Let nc, cd, jac, and foc variables denote the number of contacts, contact duration, Jacard similarity, and Foci similarity, respectively. We compute the correlation coefficients between each possible pair of these variables as shown in Table 4.1. The obtained correlation coefficients show a positive dependency between the contact pattern for a pair of nodes and their social similarity which in turn supports our hypothesis.

4.2.5 Graph Inference using Social Similarity

As we saw in the previous section, there is a dependency between the pattern of interactions among nodes and their social similarities. We can interpret the total number of contacts or the total contact duration between two nodes as their level of interaction. To construct the contact graph G, we add an edge between two nodes if they met each other at least once. We assign to each edge a weight which shows the total contact duration in the conference. People can randomly contact each other during the conference; therefore, if we include all contacts in the contact graph, we see almost a complete graph as shown in Figure 4.2.



(a) Infocom 2005 (b) Infocom 2006

Figure 4.2: Contact graphs with different threshold values

In this section, we assume that for the graph $G = (V, E)$ the edge set E is unknown; however, we have the information about social profiles of nodes in V. By having social profiles, we calculate the social similarities between all possible pairs in V by applying the three described social similarities. We store the similarity results in separate lists called L_{sim}. Next, we sort each similarity list in a decreasing order. These sorted lists are our predictor results which is going to be used to infer the edges in E.

For inferring missing links, we extract from the original L_{sim} those pairs which are similar at least as much as a prespecified threshold thr_{sim}. We store these pairs in a separate list called L_{sim}^{temp}. Let us assume that there are l_{sim} pairs in the L_{sim}^{temp}. Meanwhile, we store all observed edges in G according to our real data in another list called L_{obs}. We similarly sort L_{obs} list in a decreasing order based on the total contact duration for each pair. We extract the first l_{sim} pairs from L_{obs} and store them in a separate list called L_{obs}^{temp}. Finally, we count the number of matched pairs between L_{sim}^{temp} and L_{obs}^{temp} lists. This number shows the percentage of correct predictions for thr_{sim}. Our intuition is that larger values of thr_{sim} should predict the strong social interactions more significantly than a random predictor.

To evaluate the performance of our predictor, we use different threshold values for similarities. For each thr_{sim}, we compute the percentage of correct predictions versus the fraction of all node pairs such as (u, v) for which we have $sim(u, v) >= thr_{sim}$. The percentage of correct predictions represents the precision of our predictors ($prec = \frac{TP}{TP+FP}$). Here, TP and FP denote the number of true positives and false positives, respectively. In this chapter, we only focus on the precision results of our

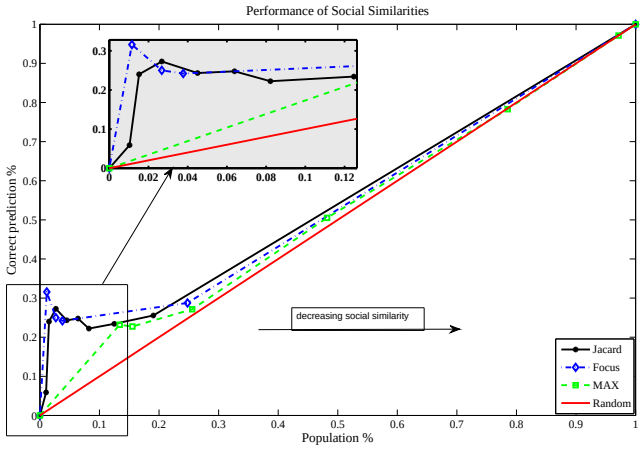


Figure 4.3: Performance of social profiles (Infocom06)

predictors. Figure 4.3 shows the results of our three social similarities. As we can see in Figure 4.3, the performances of three social similarities are statistically more significant than a random predictor. Note that the random predictor guesses the missing links completely at random. Figure 4.3 also shows that for large values of thr_{sim}, the percentage of match between our predictors and the observed edge set is higher than the random predictor. These results clearly support our social proximity hypothesis. Looking more closely at Figure 4.3, we can see that the Foci distance performs better than the Jacard and MAX especially for large values of thr_{sim}. In Figure 4.3, as we decrease the similarity threshold (i.e. using a larger percentage of the population), the effect of social profiles diminishes and our results become similar to the random predictor. This is because there are only a few number of pairs which are very close to each other. For instance, the total number of pairs which are at least 10% similar is less than 20% of all possible pairs.

For our three proposed social similarities, we assumed that all social dimensions have similar weights. However, this assumption may not hold in the real world. In a conference setting, for a given pair of nodes having the same affiliation or sharing similar interests may have higher weight on the social similarity than sharing the same country of residence. Having human contact data for a set of people with a detailed version of their social profiles will allow us to study the effect of different social dimensions on human contact patterns more carefully. We have plan to pursue this task as the future work.

4.3 Graph Inference using Contact Graph Properties

In the previous section, we showed that nodes do not meet with one another uniformly at random whereas there is a tendency for nodes to contact those ones which are socially similar to them. In this section we want to answer our second question. For a contact graph $G = (V, E)$, we randomly assign labels to vertices either as internal or external nodes. We also assume that we only have information about edges in $E_{known} \subset V_{int} \times (V_{int} \cup V_{ext})$. This means that all edges among external nodes are unknown. Our task is to infer the edges among external nodes by using the known part of the graph G. Moreover, we assume that there is not any information about social profiles of nodes.

We formulate this problem as a graph inference problem where only part of the contact graph is known. Our main intuition is to use the underlying properties of the contact graph to infer the missing edges among external nodes. We assume that for each edge in E_{known} we know the total time period that the end nodes spent together as well as the total number of times that they have contacted each other during the conference.

4.3.1 Number of Common Neighbors (NCN)

We assume that our contact graph G supports the Triadic Closure principle which is common in social networks [84]. For instance, if node u meets node v and w often, then v and w are more likely to contact each other too. Based on this principle, we count the number of common neighbors (NCN) between external nodes to compute their similarity [57]. Our intuition is that if two nodes have a large NCN, they will be more likely to interact with one another. Note that since we do not know the edges among external nodes, they can only have neighbors in V_{int}. We generate a list for all possible pairs of external nodes with the number of common neighbors for each pair. Finally, we sort the NCN list called L_{NCN} in a decreasing order. This sorted list is the output of the NCN predictor for inferring missing edges.

4.3.2 Shortest Path (SP)

It is known that social networks have low diameters [84]. By using the inverse of total time that two nodes spent together, we assign a weight to each known edge in

E_{known}. Thus, a small value for an edge weight means that the corresponding end nodes met each other frequently during the conference. For finding the similarity among external nodes, we compute the total weight of the edges in the shortest paths among them. This gives us a list of external nodes pairs with the total weight of the shortest path between them. Let us denote this list with L_{SP}. We sort this list in an increasing order. The L_{SP} list is also used for inferring the missing edges.

4.3.3 Random Walk (RW)

Motivated by the Page Rank algorithm used for finding the important web pages [11], we propose another method which explores a larger subset of paths between two nodes. Let us assume that we want to find the similarity between two external nodes u and v. We consider a random walk which starts from u and moves to a neighbor of the current node with probability of α at each step. The random walk returns to the starting node u with probability of $1 - \alpha$. This guarantees that we do not explore those parts of the graph which are far from u and v. At each step, we choose a random neighbor with a probability which is proportional to the weight of the corresponding edge. For this method, we use the total contact duration which two nodes spent together as the weight of the edge between them. Thus, at each step, the random walk is more likely to move to a node which is similar to the current node. To find the similarity between u and v, we compute the stationary probability of v for the described random walk. Finally, we generate a sorted list of external nodes by using the computed stationary probabilities (in a decreasing order). Let us denote this sorted list with L_{RW}.

We simulate a partial contact graph by randomly choosing a subset of nodes as external ones and remove the edges among them. In our simulation, we randomly pick 50% of nodes as external nodes and remove all edges among them. We use both Infocom 2005 and 2006 data for our simulations. To evaluate the performance of our predictors, we follow the same steps as in Section 4.2. By using different percentage of the whole population, we compute the percentage of matches between the lists generated by the three described methods with the sorted version of observed edges among external nodes. Note that we sort the list of observed edges among external nodes based on their total contact duration time.

Figures 4.4 and 4.5 compare the performances of NCN, SP, and RW with the random predictor. Our results show that both SP and RW predict the missing edges

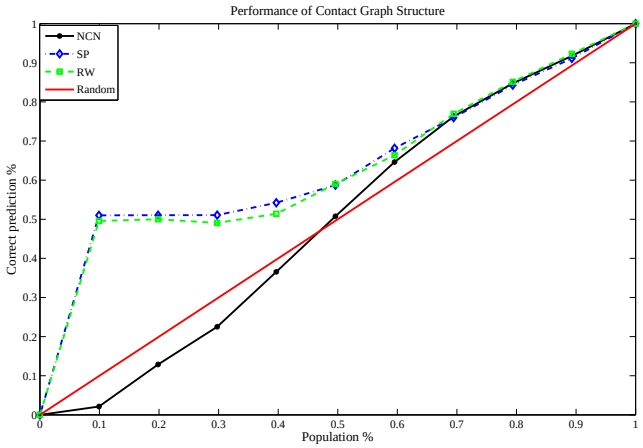


Figure 4.4: Performance of contact graph structure (Infocom05)

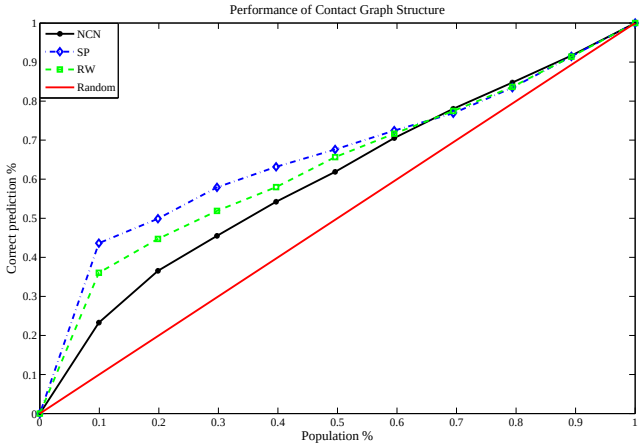


Figure 4.5: Performance of contact graph structure (Infocom06)

statistically better than the random predictor. As we see, the SP predictor outperforms others and the RW predictor has the second rank. These results highlight the importance of using the contact graph structure for inferring the missing links. This also shows that there is an underlying mechanism governing the formation of links in a contact graph that clearly cannot be explained by a purely random process. Figure 4.4 also shows that NCN does not work efficiently for Infocom 2005 data. This is because the Infocom 2005 dataset contains contact data for a limited number of participants who might have already known each others. Thus, using the number of common neighbors cannot provide useful information for inferring missing edges for this specific setting.

4.4 Contact Graph Model

In the previous sections, we saw that contact graphs have a structure which is clearly different from a random contact graph. Looking more closely at Figures 4.3, 4.4, and 4.5, we recognize a local maximum in the beginning of the graphs. In this region where chosen nodes are very similar, we see that the maximum difference between our predictors and the random predictor happens. In [84], Watts and Strogatz proposed a graph structure to model small-world networks. Their graph models have properties of both random graphs and structured graphs (e.g. a lattice). We use the same idea to model the structure of a contact graph. Let us assume that our network has N nodes. We generate a social graph for these N nodes by following the same model as Watts where we place N nodes on a one-dimensional lattice. We connect every node to its k nearest neighbors with respect to the lattice structure. Let us call these links short range links. Then, we rewire each short range link with a probability p to add random graph properties to our social graph. We call these links long range links. Both short and long range links connect a node to other socially similar nodes. The resulting social graph is used as the underlying structure by which we generate our contacts.

To generate contacts, we randomly pick a node u from all N possible nodes in our generated social graph. Next, we choose the node v which is going to meet u with a probability q uniformly at random from all N nodes. This models the fact that nodes may contact each other at random. With probability of $1 - q$, we choose the peer node for u from its neighbor set in the social graph structure. This models the fact that similar nodes are more likely to see each other. Following these steps,

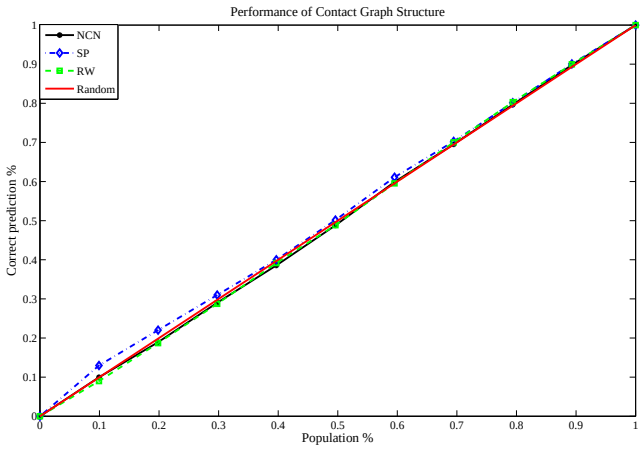


Figure 4.6: Performance of contact graph structure (synthetic mobility with $q = 1.0$)

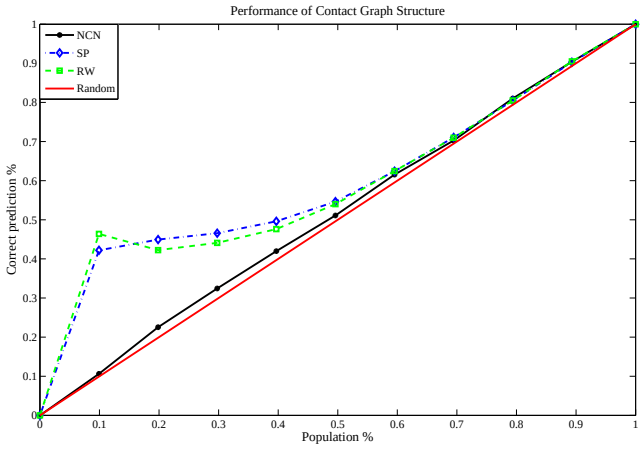


Figure 4.7: Performance of contact graph structure (synthetic mobility with $q = 0.2$, $p = 0.2$, and $k = 5$)

we generate a contact graph by having k, p, and q parameters. Intuitively speaking, larger values for q makes the resulting contact graph to be similar to a random graph whereas small values for q appreciate the homophily process in the generated contact graph.

For simulating a contact graph, we assume that our network has $N = 100$ nodes. For the first run, we simulate a contact graph with $q = 1.0$ in which nodes contact each other uniformly at random. We pick 50% of nodes as the external nodes. Figure 4.6 shows the results of NCN, SP, RW predictors. As we see, none of predictors performs better than the random predictor because there is not any structure in the simulated contact graph. For our second test, we choose $k = 5$, $p = 0.2$, and $q = 0.2$ to simulate another contact graph. The results of our three predictors are shown in Figure 4.7. Interestingly, in this case we can see similar patterns for SP and RW predictors as the ones which we observed in the real data. Thus, if nodes are more likely to see the ones which are similar to them (e.g. their neighbors on the underlying social graph), we are able to explain the observed local maximums which appears in the beginning of our predictors graphs.

4.5 Discussion

In this chapter, we formulated the problem of social contact prediction as a graph inference problem. First, we showed that we can predict the edges of a contact graph when we only have social profiles of people, but we do not have any information about the contact graph itself. Next, we studied the effectiveness of using the underlying properties of a contact graph for the contact prediction problem where only a part of the contact graph is known. In both settings, our prediction results are statistically more accurate than a random predictor. All of the proposed contact predictors are based on well known properties of social graphs. Finally, motivated by the small-world networks model, we showed that the performance of our predictors can be justified by considering the role that the homophily process plays on the structures of contact graphs.

In this chapter, we only used the contact data collected from two conferences in order to study the contact prediction problem; however, we believe that all of our contact predictors are useful for other datasets collected from large geographical spaces such as cities and campus environments. We can justify this by considering the success of our predictors in conference settings which are hard contexts for predicting

people contacts. The reader should note that in a conference people are more likely to move randomly in order to meet new people whereas in a daily life it is shown that people move more predictably due to their regular travels between home and work place [76]. Furthermore, it is also important to highlight the fact that all of our predictors are based on the homophily principle and the small-world network properties of contact graphs. Thus, we can successfully use the proposed methods in this chapter for other social settings in addition to conference environments.

Chapter 5

Predicting Missing Contacts in Mobile Social Networks

In several well documented studies [22, 14, 78, 54], researchers distributed a limited number of short range wireless sensors among a set of people to record when they are in close proximity of each other. More specifically, whenever a person u who carries a sensor device comes into the close proximity of another person v who carries a wireless sensor or a Bluetooth enabled device, person u's sensor records a *contact event* with person v. In this chapter, we focus on those experiments in which wireless sensors are carried by a set of people to collect their contact events. We represent the set of events by a directed graph called *contact graph*, where the nodes are people and the edges are contact events. We call the nodes which carry sensor devices *internal nodes* and those which carry Bluetooth enabled devices such as cellphones or PDAs *external nodes*.

In experimental datasets that we analyzed (see Table 5.1) we found that internal nodes recorded a large number of contacts with external nodes. While internal nodes can record the presence of all other nodes including internal and external ones, the external nodes are not able to detect any contact event. As a result, a large portion of the sampled contact graphs, specifically the contacts among all external nodes, is missing. Real data which were collected in various settings show that people with Bluetooth enabled devices (e.g. external nodes) by far outnumbered those with wireless sensor ones.

We are interested in reconstructing complete graphs from those partial contact graphs that were collected in a real experiment. We formulate the problem of inferring

missing part of a contact graph as a contact prediction problem, and we propose several methods for predicting the missing contacts. Our proposed methods predict missing contacts among external nodes by exploiting the underlying properties of contact graphs. Predicting missing contacts will complete our observations about different statistical features of partial contact graphs. In Section 5.3, we particularly show that predicting missing contacts allows us to restore communities information which otherwise are missing.

In this chapter, we study a variety of contact traces collected from different social settings such as [22, 14, 78, 54] for our analysis. Based on the observed time-spatial properties of contact graphs, we propose three different methods to make contact predictions by computing similarities between neighbor sets of external nodes. We also investigate the effectiveness of using nodes' contact rates for predicting missing contacts. Furthermore, it was previously shown that the contact probability between mobile wireless devices is influenced by their owners' social characteristics [22, 65]. We call the set of social characteristics for each user her *social profile*. In this chapter, we employ two socially-based methods for predicting missing contacts, and study their performance using a contact trace collected from a conference setting [14].

Our results show that we can reliably reconstruct missing parts of contact graphs by using the proposed methods. This enables researchers to expand the existing collected contact traces in order to include the contacts among external nodes as well. Our solution to the contact prediction problem also sheds light on the way in which people move. While the problem of link prediction is not new in the context of social networks [57, 27], to the best of our knowledge this work is the first one that tries to address this problem in the context of *Mobile Social Networks*, that is, a network consisting of contacts among a set of mobile users. The results of this chapter was published as two papers in the World of Wireless Mobile and Multimedia Networks conference [49] and the Pervasive and Mobile Computing journal [46]. The main contributions of this chapter can be summarized as follows:

1. We present the problem of contact prediction in the context of mobile social networks and show how we can study this problem by using real data from different social settings.

2. We propose several methods each of which makes use of one of the time-spatial, popularity, or social information to reconstruct missing part of a contact graph.

3. Finally, we integrate social information with time-spatial information to propose a more effective method for contact prediction.

5.1 Problem Definition

A contact event between two users u and v can be shown by a quadruple (u, v, t_s, t_e) implying that user u's device detected user v's device in close proximity in the $[t_s, t_e]$ time interval. We assume that every human contact between u and v is recorded as a contact event by one of the sensors carried by u or v. It is important to note that not every observed contact between two devices necessarily means a social interaction between people who carry the devices. For the rest of this chapter, we only focus on analyzing those contacts that are collected by wireless sensors in an experiment.

We show all contacts recorded by internal nodes during an experiment using an undirected contact graph $G = (V, E_{known})$. Reader should note that there is an undirected edge between nodes u and v if at least one of them recorded the presence of the other one in its proximity. Here, we denote the set of all people participating in the experiment with $V = V_{int} \cup V_{ext}$ where V_{int} and V_{ext} are the sets of internal and external nodes, respectively. We assume that $|V| = N$ and $|V_{int}| = N_{int}$ where $N_{int} < N$. We also denote the set of known edges by E_{known} where we represent every observed contact such as (u, v, t_s, t_e) with an undirected edge from node u to node v in G. From the data provided by experiments such as those listed in Table 5.1, we can only construct a partial contact graph. In other words, while the set of edges in $E_{known} \subset V_{int} \times (V_{int} \cup V_{ext})$ is known, all edges between external nodes are missing. Our problem is to infer missing edges among external nodes by using the available information from the known part of the graph (e.g. E_{known}), that is, to predict the edges in $V_{ext} \times V_{ext}$ which would have been detected if the nodes in V_{ext} had carried wireless sensors.

5.2 Reconstructing Contact Graphs

In this section we describe the essential properties of contact graphs that are useful for contact prediction. We also propose three different sets of prediction methods based on the underlying properties of contact graphs. Finally, we explain our main algorithm that makes use of our proposed methods to infer missing contacts among external nodes.

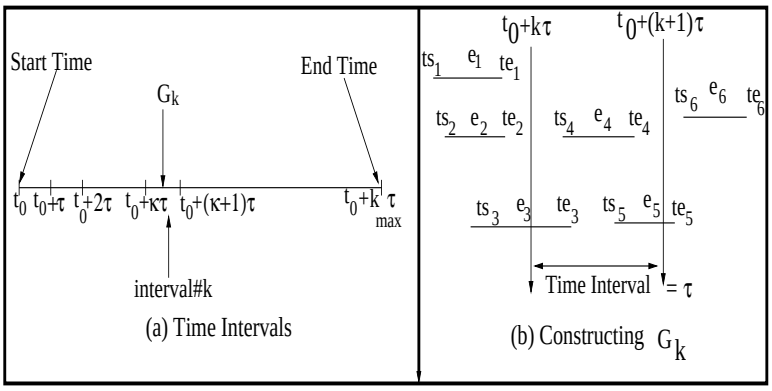


Figure 5.1: Constructing the partial contact graph G_k

5.2.1 Constructing Partial Contact Graphs

Since the collected contact traces change over time (e.g., edges between nodes appear and disappear), we divide the experiment time into equal intervals of τ seconds called *time intervals*. We choose $\tau = c \times T$ where c is a constant integer, and T is the *inquiry interval* of wireless sensors that is the time gap between two consecutive sensings. The coefficient c should be chosen carefully according to the dataset setting. Usually c is chosen to be either 1 or 2. Let $\Lambda_k = [t_0 + k\tau, t_0 + (k+1)\tau]$ denote the k^{th} time interval where $0 \leq k < k_{max}$ and t_0 is the starting time of the experiment (see part (a) of Figure 5.1). For each time interval Λ_k, we construct a contact graph G_k by collecting all contacts that were observed by internal nodes in Λ_k. We show the k^{th} contact graph with $G_k = (V, E_k^{known})$ where $V = V_{int} \cup V_{ext}$ is the set of all nodes and E_k^{known} is the set of all known edges of G_k (e.g. observed contacts by the internal nodes). In Figure 5.1, G_k contains all three contacts e_3, e_4, and e_5 that are observed by internal nodes in Λ_k. Our goal is to predict missing edges $E_k^{unknown}$'s by exploiting the information about the known edges of G_k's where $E_k^{known} \subset V_{int} \times (V_{int} \cup V_{ext})$ and $E_k^{unknown} \subset V_{ext} \times V_{ext}$.

5.2.2 Contact Graph Properties

There are three elements that play essential roles in the contact process: (1) time-spatial locality, (2) social similarity, and (3) popularity. Next, we discuss the importance of these elements in the structure of contact graphs.

Time-Spatial locality: A contact between two nodes u and v at time t means that u and v were in close proximity of each other at time t. If node u records a

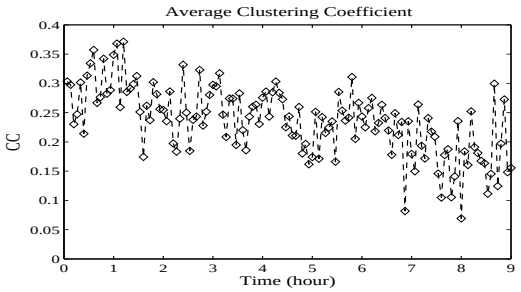


Figure 5.2: The average clustering coefficient (Infocom06: 9:00 AM to 6:00 PM)

contact with node v at time t and node u also records a contact with node w at time $t + \delta$ for a small δ, we can assume that nodes v and w are in a close distance of each other around time t. If two nodes are geographically close, we expect that they are more likely to meet each other in the near future.

Social similarity: Let $sim_{soc}(u, v)$ denote the social similarity between two nodes u and v. We assume that if node u is more socially similar to v than w (e.g. $sim_{soc}(u, w) < sim_{soc}(u, v)$), then u is more likely to contact v than w.

Popularity: Popularity in social networks is captured by nodes' degrees. Let us define a node's *contact rate* in time interval Λ_k as the number of contacts a node made during Λ_k. A node's contact rate in a mobile social network is similar to a node's degree in a social network, in that it reflects the social role of the device's owner. For example, if node u's owner is a conference organizer, it probably has a high contact rate. Thus, if node u has a higher contact rate than v, we assume that node u has a higher probability to contact a given node w than v does.

5.2.3 Methods Based on Neighborhood Similarity

To predict contacts, one approach is to exploit the underlying properties of contact graphs. Here we want to show that contact graphs have a neighborhood-cohesiveness property in which neighbors of a given node have a high probability of being connected to each other. First, we construct the contact graph G_k's for all time interval Λ_k's as described earlier. The clustering coefficient of node u in G_k is the fraction of pairs of u's neighbors that are connected to each other by edges [84]. We compute the average clustering coefficient of the contact graph G_k denoted with CC_k, by taking the average of clustering coefficients of all nodes in G_k. In Figure 5.2, we show the average clustering coefficients for the contact graphs during the second day of Infocom 2006 where $\tau = 2 \times T = 4$ minutes. Interestingly, the total average of all computed

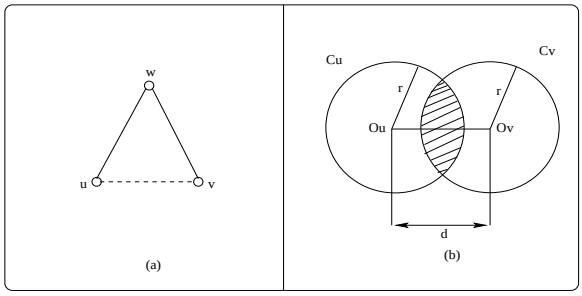


Figure 5.3: The effect of common neighbors on geographical proximity

CC_k values is shown to be around 25%.

To justify the neighborhood-cohesiveness property of contact graphs, let us assume that we are running an experiment with N_{int} sensors when all of them have similar sensing ranges of r. Moreover, suppose our nodes are uniformly distributed in the experimental region (e.g. a conference hotel). In this graph, there is an undirected edge between two nodes u and v in Λ_k if their distance is less than r in that time interval (e.g. $d(u, v) \leq r$). Now, let us consider the simple scenario shown in part (a) of Figure 5.3 where two nodes u and v detected a common neighbor w at the same time. The existence of links (u, w) and (v, w) implies a geographical proximity between u and v which in turn increases the chance that u and v also sense each other. It is not hard to see that the number of common neighbors (NCN) between two given nodes u and v is proportional to the intersection area between their radial circles as shown in part (b) of Figure 5.3 in which O_u and C_u represent the location of node u in the experimental region and u's radial circle, respectively. A large NCN for two given nodes implies a large intersection area between their radial circles which in turn indicates a geographical closeness between them.

Now, we discuss three methods that measure the geographical closeness between two nodes based on the intersection between their neighbor sets. In all the following equations, $sim^k(u, v)$ denotes the similarity between two nodes u and v in Λ_k, and $N^k(u)$ represents the neighbour set of node u which is the set of contacted nodes by u in time interval Λ_k.

$$sim^k_{ncn}(u, v) = \left|N^k(u) \cap N^k(v)\right| \tag{5.1}$$

$$sim^k_{jac}(u, v) = \frac{\left|N^k(u) \cap N^k(v)\right|}{|N^k(u) \cup N^k(v)|} \tag{5.2}$$

$$sim^k_{min}(u,v) = \frac{\left|N^k(u) \cap N^k(v)\right|}{min(\left|N^k(u)\right|, \left|N^k(v)\right|)} \tag{5.3}$$

While Equation 5.1 simply calculates the *NCN* between a pair of nodes to estimate their closeness, Equations 5.2 and 5.3 are the normalized versions of the NCN method. Equation 5.2 (*Jacard* method) makes use of the Jacard index to measure the similarity between the neighbor sets of the given nodes [43]. To clarify the main difference between Equations 5.2 and 5.3 (*Min* method), let us consider two scenarios. In the first scenario, let us assume that two nodes u and v contacted two similar nodes in time interval Λ_k. Also suppose node u only contacted these two nodes while node v contacted ten nodes including these two common nodes in Λ_k. We may desire a higher significance for the described scenario than if each of u and v met six nodes in Λ_k where two of these six nodes are in common. Although the Jacard index gives $\frac{1}{5}$ for both scenarios, the Min method assigns a higher score to the first scenario [27].

5.2.4 Methods Based on Social Similarity

Considering the importance of the homophily principle in the link formation process in social networks [61], we want to test the power of social similarity for predicting contacts among nodes in a mobile social network. In [48] we studied the influence of the similarity of people's social profiles on their contact probabilities in a conference environment. Eagle et al. [22] and Mitbaa [65] also found a close relation between human mobility and their friendship networks. In this chapter, we have access to brief social profiles of people who attended Infocom 2006 conference. In these social profiles, people reported their social characteristics. Social characters are classified according to different social dimensions such as affiliation, interest, country of birth and so on. Readers can find the details of the Infocom 2006 dataset in Chapter 3.

Jacard Social Similarity

By employing the Jacard index, we compute the total social similarity between two nodes u and v as described in Equation 3.2.

Foci Social Similarity

Another method to compute the social similarity between two nodes is by employing the Foci similarity described in Equation 4.2 of Chapter 4.

5.2.5 Method Based on Popularity

Motivated by the Preferential Attachment model for social networks [8] in which a node u connects to another node v with a probability that is proportional to v's degree, we assume that the contact probability between two nodes in a mobile social network depends on their individual contact rates. Let λ_u^k denote the contact rate of node u in k^{th} time interval that is, the number of its contacts in Λ_k. We assume that the combined contact rate between two nodes u and v is proportional to the product of their individual contact rates. Thus, we define the popularity measure between two nodes u and v in time interval Λ_k as below:

$$sim_{pop}^k(u,v) = \lambda_u^k.\lambda_v^k \tag{5.4}$$

To compute node u's contact rate we count the number of u's contacts in time interval Λ_k. In Section 5.3, we employ all six Equations 5.1, 5.2, 5.3, 3.2, 4.2, and 5.4 as our proposed prediction methods to reconstruct the missing parts of contact graphs.

5.2.6 Reconstruction Algorithm

The following steps describe our algorithm for reconstructing the missing parts of G_k's by selecting one of the previously presented prediction methods:

1. First, we generate partial contact graph G_k's for all k values as we described in Subsection 5.2.1.

2. Next, we compute the similarity scores between all pairs of external nodes by using one of our prediction methods. These similarity scores basically estimate the contact probabilities among external nodes. Therefore, for each time interval Λ_k we obtain quadruples such as $(u, v, k, sim(u, v))$ where u and v are external nodes, $sim(u, v)$ is the computed similarity score for nodes u and v, and k is the time interval number. We store all quadruples whose similarity scores are greater than zero in a similarity list denoted by L_{sim} for the post-processing step. We repeat the same process for all intervals ($0 \leq k < k_{max}$) and store all the quadruples in the same list.

3. When we finish with all intervals, we sort the L_{sim} list in a descending order based on the computed similarity scores. The sorted version of the similarity list is our predictor results.

Table 5.1: Real Data Description

Dataset	Inf 05	Inf 06	MIT	Camb	Roller
Mobile nodes	41	79	97	36	62
Length	3 days	4 days	246 days	11 days	3 hours
Scanning period	120 sec	120 sec	300 sec	600 sec	15 sec
External no	206	4321	20698	11367	1050
Total contacts	227657	28216	285512	41587	132511
Ext. contacts	57056	5757	183135	30714	72365
Ext. contacts %	25%	20%	64%	74%	55%

4. To infer the missing contacts, we select the first *Rank* number of predictions from the sorted list of L_{sim}. For each prediction quadruple, we create an edge between the corresponding external nodes.

5.3 Performance Evaluation

In this section we evaluate the performance of all proposed methods in the previous section by using the available datasets. Our ultimate goal is to see which method is more effective for predicting missing contacts.

5.3.1 Real Data Description

Here, we are planning to use contact traces collected from four different social settings. Table 5.1 describes these datasets. Infocom 2005 and Infocom 2006 datasets were collected from Infocom conferences in 2005 and 2006 respectively. Participants in Infocom 2005's experiment belong to different social communities; however, in the Infocom 2006 participants were especially selected such that 34 people out of 79 were from four research groups [39]. In Infocom 2006 dataset, participants also reported a brief version of their social profiles as was described in Chapter 3 [14]. In Cambridge dataset (Camb), the wireless sensors were distributed mainly between two groups of undergraduate students from University of Cambridge, and some graduate students from a research lab [54]. Note that both Infocom 06 and Cambridge datasets also include data for stationary wireless sensors; however, in this chapter we only use the sampled contacts by mobile sensors. While Rollernet dataset (Roller) contains the contacts from a set of people who participated in a rollerblading tour in Paris [78], Reality dataset (MIT) includes contact data of students and staff at MIT for a period

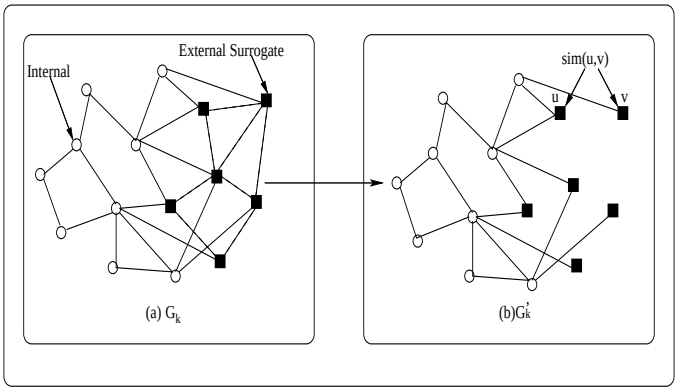


Figure 5.4: Simulating a partial contact graph

of 9 months [22].

As we see in Table 5.1, the number of external nodes in all datasets are much larger than the number of internal nodes. Moreover, the ratios of sampled contacts with external nodes to total contacts are significant. This observation is our main motivation to propose a prediction algorithm for reconstructing missing parts of partially sampled contact traces. The reader should note that the numbers of external nodes shown in Table 5.1 for different datasets are upper bounds for the unique number of people who carried their Blutooth-enabled devices. For instance, one person may carry multiple Bluetooth devices such as cellphone, mouse, headphone and so on at the same time. Therefore, it would be important to compute the real number of people who carried wireless devices in order to extract meaningful contacts from each dataset.

One algorithm for estimating the unique number of devices is to bucketize all existing devices in the dataset based on the total contact duration times that they were seen by all sensors. Thus, we store all nodes which were seen for almost the same total amount of time in the same bucket. As the first try, we can approximate the unique number of people in the experiment by returning the number of non-empty buckets. We can also achieve better approximation by going through each bucket's nodes and check how many times they were seen together in the dataset.

5.3.2 Testing Reconstruction Algorithm using Real Data

Since external nodes do not carry sensors, there is no way to validate the inferred edges between them. Instead, we choose a random subset of internal nodes and label them as *surrogates* of external nodes. These surrogates act as external nodes.

We remove all recorded contacts between surrogates from our G_k's. This process is shown in Figure 5.4 where the surrogates are shown as squares in the original graph G_k before edge removal. To generate the partial graph G'_k, we remove all contacts that were recorded by surrogates, but we keep the contacts recorded by the remainder of internal nodes (i.e. circle nodes in Figure 5.4). These partial G'_k's are the inputs for our reconstruction algorithm.

To validate an inferred contact, we examine the original contact trace that includes all contacts among all internal nodes to see if we can find a match. For example, for an inferred contact such as $(u, v, k, sim(u, v))$, we search through our complete dataset to see if there were any contacts between u and v in the $[t_0 + k\tau, t_0 + (k + 1)\tau]$ time interval. We are able to do this because the surrogates are actually internal nodes and we have all their contact data.

5.3.3 Why Predicting Missing Contacts?

When raising an issue on the missing links, it is important to motivate the research community by showing an example that at least one statistical feature extracted from experiments requires significant modification when missing links are additionally considered. To address this issue, we focus on the contact data collected from the first day of Infocom 2006 conference. Since we do not have any information about contacts between real external nodes, we only consider the contacts between sensors. We choose 75% of the sensors from the Infocom data at random and label them as external surrogates. We remove all recorded contacts between those external surrogates in order to generate a sampled contact graph. The randomly selected surrogates play the role of cellphones for us. Next, we construct a weighted contact graph by taking all nodes which were seen in the first day of the conference. We add an edge between two nodes if and only if at least one of them is labelled as sensor and recorded at least one contact with the other node during the event. We assign a weight to each edge in order to represent the total time that the corresponding end nodes were in proximity of each other. As a result, we remove all edges between the selected external surrogates in the sampled contact graph. If we assume that there is an optimal predictor which perfectly predicts missing edges between external surrogates, we are able to evaluate the value of adding the missing edges back to the sampled contact graph in order to reconstruct the original contact graph.

We compare the probability distributions of the number of contacts, contact du-

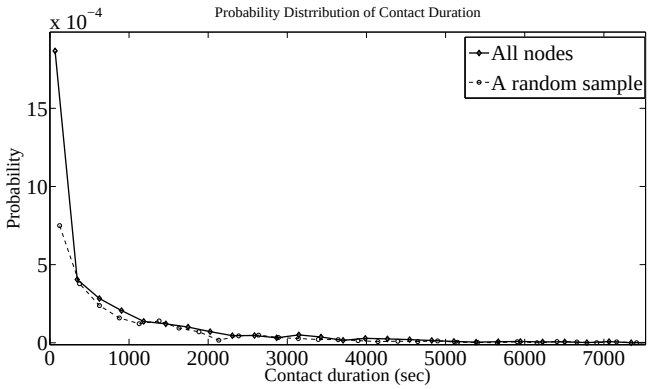


Figure 5.5: Contact duration distribution

ration, and inter-contact time between all pairs of nodes in the original contact graph that contains all contacts with the sampled contact graph that does not contain all contacts. Figure 5.5 shows the probability distribution of the contact duration between all pairs of nodes in the original and the sampled contact graphs. As we see, both distributions follow the same pattern. We also observed the same pattern in the probability distributions of number of contacts and inter-contact time. In other words, if we focus on the probability distributions of the three features above, we do not observe a significant difference between the original contact graph and its sampled version. Let $G = (V, E)$ and $G' = (V, E')$ denote the original and the sampled contact graphs, respectively. Let S also denote the set of sensors in the sampled contact graph. We assume $|S| = \frac{n}{4}$ where $|V| = n$. Thus, the probability that an edge from the original contact graph does not appear in the sampled contact graph is the probability that both end nodes are selected as external surrogates, that is $\frac{3}{4} * \frac{3}{4} = \frac{9}{16}$. Now, we compute the probability that a given triangle from the original contact graph appears in the sampled contact graph as follows:

$$Pr(\Delta) = (\frac{1}{4})^3 + 3 \times (\frac{1}{4})^2 \times (\frac{3}{4}) \approx 0.15 \tag{5.5}$$

Equation 5.5 shows the probability that a triangle from the original contact graph appears in the sampled version is around 15% when the size of selected sensors is 25%, which is relatively high. Losing triangles information from the original graph means that we lose communities' information if we do not predict missing edges. The contact duration is considered as a good metric to measure the strength of a relationship between two people. We take different threshold values for contact duration between two nodes and filter out all edges whose weights are less than the

Table 5.2: Number of *k-cliques* (original/sampled contact graphs)

k	3	4	5
$thresh = 3600$ **sec**	3304/35	90/0	6/0
$thresh = 7200$ **sec**	20/0	0/0	0/0

Table 5.3: The percentage of missing part of contact traces

Dataset	Inf 05	Inf 06	MIT	Camb	Roller
Edge Loss %	52%	56%	61%	56%	55%

threshold. We use 3600 and 7200 seconds as our thresholds. After filtering out the edges whose weights are below the threshold, we count the number of *k-cliques* which are complete graphs on k nodes in the original and sampled contact graphs. The k-cliques represent communities in contact graphs. Our results are shown in Table 5.2. We can see that the number of k-cliques in the sampled contact graph is much less than the original graph. Therefore, we can conclude that predicting missing contacts returns missing triangles and communities back to the partial sampled graph.

5.3.4 Contact Prediction using Time-Spatial and Popularity Information

To evaluate the performance of our prediction methods, for all our datasets we randomly choose 75% of internal nodes and label them as surrogates. We then construct the partial contact graph G'_k's as described earlier. The resulting partial graphs include only a small subset of nodes. Table 5.3 shows the average percentage of contacts that we discard by labeling 75% of the internal nodes as surrogates. Our goal is to infer the missing contacts between the surrogates of the partial contact graph G'_k's. Note that for the Infocom 2005 and 2006 datasets, we use only the collected contacts on the first day of the main conference because the number of participants were maximum on this day; however, for the Cambridge and Rollernet datasets we use all contacts for our analysis. We also use 35 days of the MIT data for our analysis. We repeatedly run our prediction algorithm with 20 different random sets of surrogates, and the presented results are the average values along with their 95% confidence intervals.

Let us first focus on the performance of methods that are based on neighborhood similarity and popularity. To infer the missing contacts, we pick the first *Rank* number of predictions from our sorted L_{sim} list as the most confident predictions.

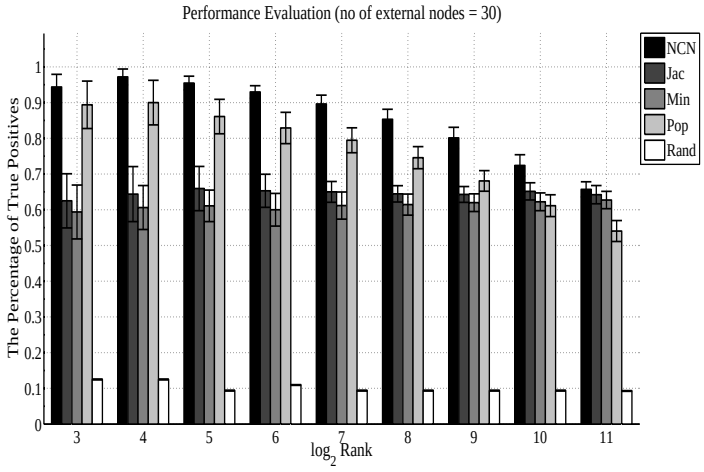


Figure 5.6: Percentage of true positives for contact predictions (Infocom 2005)

Basically *Rank* is the position of a quadruple prediction in the sorted version of L_{sim}. Next, we compute the *precision* or the percentage of matches (i.e. *true positives*) between our prediction results and the real data by inspecting our database. We increase the *Rank* value to see how different methods operate as we increase the number of predictions. For comparison purposes, for our baseline we use a simple *random predictor* that randomly selects a pair of surrogates and a time slot for a prediction. In the following figures the X axes show the logarithm of the *Rank* instead of the *Rank* itself in order to not show big numbers. Hence, the $\log(Rank) = 3$ means that we take the first 8 quadruples from the sorted L_{sim} and test them to find out what percentage of those contacts actually happened in the real data. Figures 5.6, 5.7, 5.8, 5.9, and 5.10 show the percentages of true positives for the Infocom 2005, Infocom 2006, Cambridge, Rollernet, and MIT datasets, respectively.

From the given figures, we make several important observations. First, the NCN, Jacard, Min, and Popularity predictors significantly outperform the random one, proving that there is indeed useful information even in partial contact graphs which can be used for prediction purposes. Second, it is evident that in most of our evaluations, the methods based on the neighborhood similarity perform better than the popularity method which highlights the importance of using the time-spatial locality for predicting missing parts of contact traces. The reader should note that the popularity method does not contain any location information, unlike the neighborhood similarity methods. Third, as we increase the *Rank* value, the percentage of true positives drops, while the percentage of false positives increases. This is because we

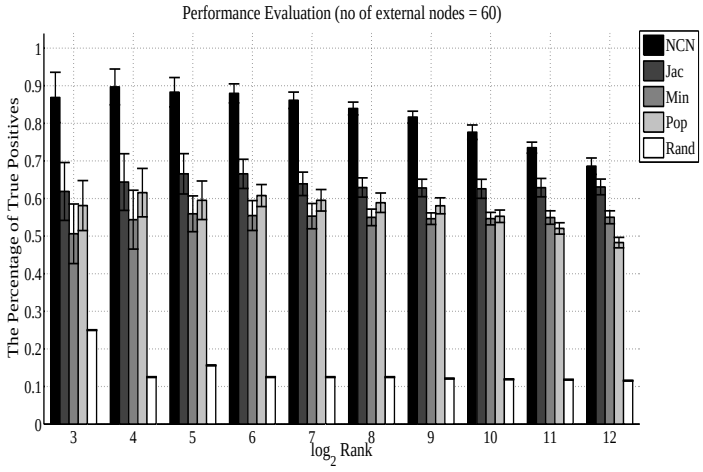


Figure 5.7: Percentage of true positives for contact predictions (Infocom 2006)

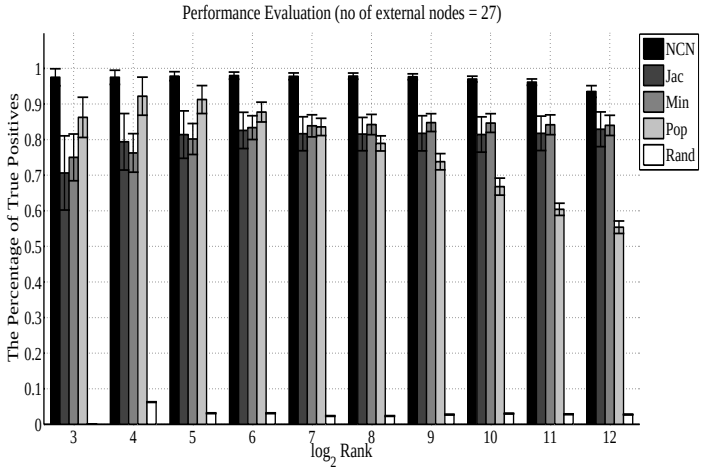


Figure 5.8: Percentage of true positives for contact predictions (Cambridge)

sort our similarity list in a descending order.

Interestingly, in the Rollernet dataset we observe that the effect of popularity drops more significantly than other datasets as we increase the *Rank*. To justify this we should recall the social structure of people who participated in this experiment. One group consisting of 25 staff members were asked to stay at previously assigned positions in front and back of the tour. There was another group of friends with 11 nodes as well as a group of skilled skaters with 26 nodes. We believe that most of these people except skilled skaters were not very mobile and were located most of the time in the same relative position inside the tour. As a result, the performance of the popularity method without location information drops faster than that of a

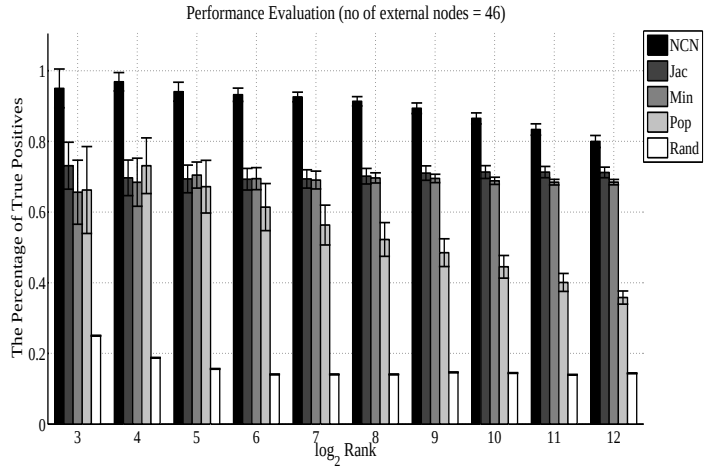


Figure 5.9: Percentage of true positives for contact predictions (Roller)

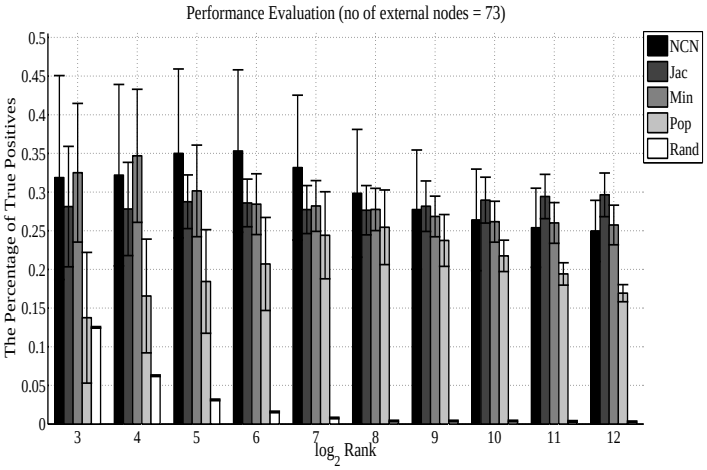


Figure 5.10: Percentage of true positives for contact predictions (MIT)

conference setting in which people have a chance to meet each other at least every two hours during coffee breaks.

Furthermore, results from the MIT dataset show that for large geographical spaces (e.g. campus environments) the percentage of true positives for contact predictions is low. Let us define the density of a contact graph G_k as below:

$$D_k = \frac{2|E_k|}{|V|(|V|-1)}, \tag{5.6}$$

where E_k is the observed edge set of contact graph G_k in time interval Λ_k and V is the set of all sensors in the experiment. To explain the low performance of our

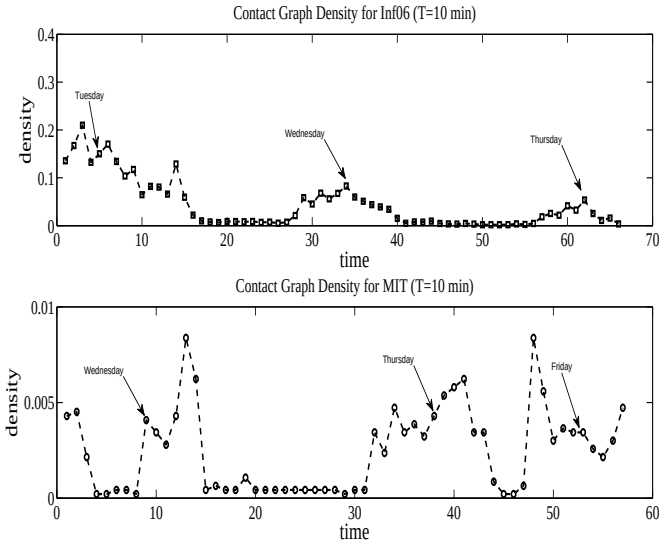


Figure 5.11: Evolution of Contact Graph Densities for MIT and Infocom 2006 datasets

predictors for the MIT dataset, we compute the density of contact graphs for a period of three days for the MIT and Infocom 2006 datasets. In both settings, we use the complete data of all sensors for our analysis. We also choose a time interval equal to $T = 10$ minutes to construct the contact graphs.

Figure 5.11 shows the evolution of contact graph densities for three consecutive days for both MIT and Infocom 2006 datasets. As we can see, the density of MIT's contact graphs are much lower than the Infocom 2006's. In particular, the average density of MIT's contact graphs is 18 times less than Infocom 2006's. We can explain this by considering the fact that in a conference setting people are very likely to see each other frequently during the conference whereas in the MIT dataset people are less likely to contact each other very often because of the nature of daily life and the large size of the environment. As a result, in large geographical spaces such as the MIT dataset it is likely to have a subset of external nodes where there are no internal nodes or just few number of internal nodes in their proximity. Therefore, the geographic based methods fail to predict the missing contacts among such external nodes and different prediction methods may have to be devised.

It is necessary to mention that most real contact traces are quite noisy, and in particular they can miss many contacts. Authors of [69] listed several issues related to iMotes software which were used in the the Infocom, Cambridge, and Rollernet experiments. The reset issue because of memory overflow, the synchronization issue

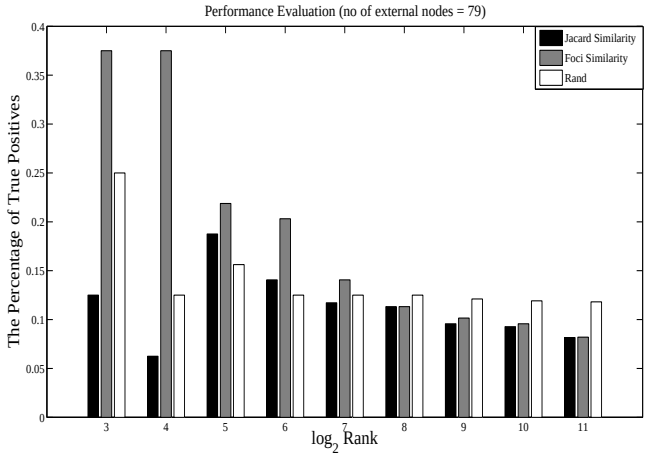


Figure 5.12: Percentage of true positives for contact predictions using social data (Infocom 2006)

because of random seeds, and the limitation for the number of responses returned in a Bluetooth scanning caused by the Bluetooth protocol stack are the most important ones. All of these issues can cause iMotes to miss some of real contacts. Thus, the number of computed false positives for our predictors may be overestimated because some of those false positives could have actually happened in the reality, but iMotes failed to capture them.

5.3.5 Contact Prediction using Social Information

As we have mentioned earlier, the Infocom 2006 data also includes participants' social profiles. In this part of our analysis we assume that we do not know anything about the contact trace except the social profiles of participants. In other words, we assume that all internal nodes act as surrogates of the external nodes. For testing our social methods, we compute social similarities between all possible pairs of surrogates (e.g. $(u, v, sim(u, v))$ by using Equations 3.2 and 4.2 independently. We store the computed Jacard and Foci similarities in L_{sim}^{jac} and L_{sim}^{foc} lists, respectively. We sort both of these similarity lists in descending order. These two lists are our social predictor results for the Jacard and Foci similarities. To validate a prediction based on social similarity such as $(u, v, sim(u, v))$, we randomly select a time interval Λ_k as the time step when a contact happened between nodes u and v. For evaluation, we choose the first $Rank$ number of predictions from our sorted similarity lists and inspect them by using the Infocom 2006's contact trace data.

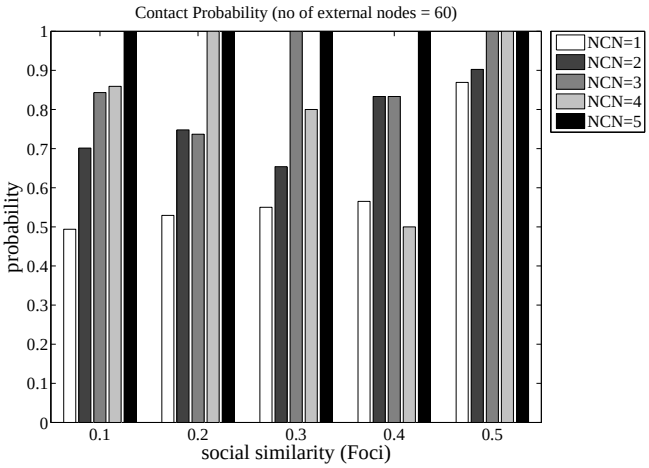


Figure 5.13: Contact probability as a function of social and proximity information (Infocom 2006)

Figure 5.12 shows the percentages of true positives regarding our prediction results when we only use social profiles. The figure shows that Foci similarity better reflects the similarity among people who attend a conference than Jacard does. From Figure 5.12, we make the interesting observation that using social data without any time and proximity information is still helpful for predicting missing contacts. The reader should note that the collected social profiles were only partial in that some people did not report their complete profiles, or any at all. By testing the distribution of social profiles, we found that there are only around 100 pairs of nodes which are socially very similar (e.g. $sim^{soc}_{foc} >= 0.2$) while the majority of nodes are not. Therefore, we expect that for $\log_2 Rank$ values greater than 7, social profiles lose their effect for prediction task. For the rest of our analysis we only use the Focus method to measure the social similarity between nodes.

We have already seen that the NCN method outperforms others as it contains the proximity information. Now, the question is if we can propose a better predictor by using both social profiles and NCN information. One could make the case that once two users are in relative proximity (e.g. in the same room), the probability of meeting each other is high if they are also friends. To characterize the effects of social focus and NCN on contact probability, we select 75% of internal nodes as surrogates and repeat our evaluations as before by computing NCN scores between surrogates. We filter those quadruples that exactly have c nodes in common (e.g. $sim^{k}_{ncn}(.,.) = c$) and store all of them in L^{c}_{sim} list. Next, we use data binning to categorize all L^{c}_{sim}'s

quadruples into five equally sized intervals based on Foci similarity. We choose our intervals as $thr_{soc} \leq sim^{soc}_{foc}(u, v) \leq thr_{soc} + 0.1$ where $thr_{soc} \in \{0.0, 0.1, 0.2, 0.3, 0.4\}$. We then compute the percentage of surrogates that actually met each other. This gives us the contact probabilities for different social similarity and NCN values. We need to repeat the same process for all possible NCN values (e.g. $1 \leq c \leq 5$).

Figure 5.13 shows the contact probability among surrogates as a function of NCN and Foci similarity. We observe that for NCN values of 1, 2 and 3, as Foci similarity between two given nodes increases, the contact probability also becomes greater. Figure 5.13 also shows that we cannot expect any improvement by adding social information when NCN is large. This is because for these cases the NCN acts as a dominant factor in contact probability. Our results are encouraging as they provide incentive to incorporate social information with the NCN method to achieve a better performance.

One possible way for combining social information with NCN would be to compute prediction quadruples by using the NCN method. We then sort all quadruples in a descending order based on their NCN scores. Next, we use the social Foci similarity as the second dimension to rank all quadruples with the same NCN scores in a descending order. This second ranking would give a higher weight to those pairs that are socially closer. This two-fold sorting would provide a better performance than when we use only one of the NCN or Foci similarity. Incorporating nodes' contact rates with NCN information with a similar approach would also enhance the performance of the NCN predictor.

5.3.6 Statistical Analysis of Contact Predictors

In the previous subsections, we analyzed our predictors with respect to true/false positives (i.e., what percentage of predicted contacts actually happened). However, an analysis of true/false negatives (i.e., what percentage of contacts that actually happened are we able to predict) is missing. In this subsection, we compute several statistical measures in order to give a full view about the performance of our predictors. To take into account both TP/FP and TN/FN, we compute five statistical measures including the true positive rate or recall (i.e. $TPR = \frac{TP}{TP+FN}$), the false positive rate (i.e. $FPR = \frac{FP}{FP+TN}$), the precision (i.e. $precision = \frac{TP}{TP+FP}$), the accuracy (i.e. $accuracy = \frac{TP+TN}{P+N}$), and the root mean square error (RMSE) for our predictors.

For each statistical measure, we determine missing contacts among external surrogates as our observations. We also compute prediction results for each method in a similar manner as we did in the previous subsection. Having observations and prediction results, we can compute TP, FP, TN, and FN numbers for different *Rank* values. Then, we can compute the first four statistical measures. We also employ the RMSE to measure the difference between values predicted by our predictors and values actually observed from the real data. The RMSE analysis besides the other four statistical measures better illuminates the relative performance of different predictors. Our ultimate goal is to better understand the predictor's performances by computing the above statistical measures while changing the *Rank* value.

For our statistical analysis, we focus on the contacts of the first day of Infocom 2006 conference from 11:00 AM to 2:00 PM. This three hours period covers different types of events in the conference. We only use three hours of Infocom data because we want to limit the number of missing contacts as well as the *Rank* value. We choose 75% of nodes and label them as external surrogates. We repeat our experiment 20 times and report the average results with their 95% confidence intervals. In Infocom 2006 dataset, we found that in average there are around $7k$ edges among external surrogates when we randomly select 75% of nodes as external surrogates. Thus, we have $3 \leq \log(Rank) \leq \lceil \log 7k \rceil = 13$.

After removing edges between external surrogate pairs, we construct all partial G_k's where $k_{min} \leq k \leq k_{max}$. We also choose $\tau = 240$ seconds as the time interval. Let V_i^{obs} denote the observation vector for the i^{th} run ($1 \leq i \leq 20$) in which we store observed edges from the real data. Thus, for each external node pair (u, v) and $k_{min} \leq k \leq k_{max}$, we set $V_i^{obs}(u, v, k) = 1$ if we observe an undirected edge between those external nodes at k^{th} time interval; otherwise, $V_i^{obs}(u, v, k) = 0$. We also present our prediction results with vector $V_{r,i}^{pred}$ for i^{th} run when $Rank = r$. In particular, $V_{r,i}^{pred}(u, v, k) = 1$ if we find the quadruple $(u, v, k, sim(u, v))$ in the first r number of predictions in the sorted similarity list; otherwise, $V_{r,i}^{pred}(u, v, k) = 0$.

Having partial G_k's, we generate the corresponding similarity list L_{sim} and sort it in a descending order for all prediction methods in each run. Next, we store the first r number of predictions from L_{sim} list in $V_{i,r}^{pred}$ vector for post-processing steps. By having observation and prediction vectors for each Rank value, we compute the average of TPR, FPR, accuracy, and precision along with their 95% confidence intervals. Let $|V_{ext}|$ denote the number of external surrogates in our experiment. Let also denote the set of all time intervals for our analysis with $\Lambda = \{k | k_{min} \leq k \leq k_{max}\}$.

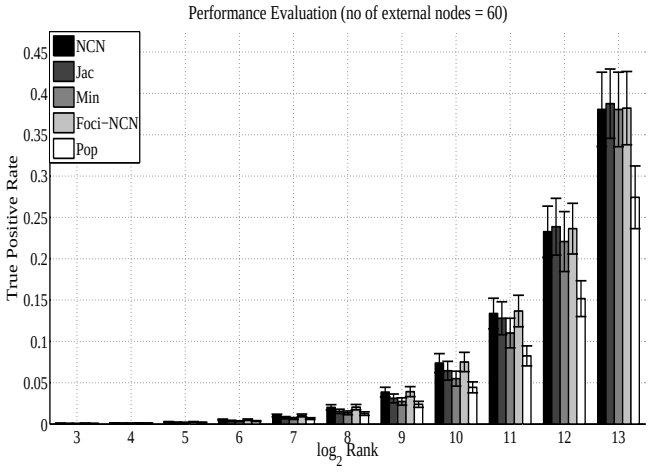


Figure 5.14: True positive rates of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06)

For a given $Rank = r$ and the i^{th} run, we compute the RMSE of $V_{r,i}^{pred}$ with respect to the observation vector V_i^{obs} as shown in Equation 5.7:

$$RMSE(V_{r,i}^{pred}, V_i^{obs}) = \sqrt{\sum_{(u,v,k)\in V_{ext}\times V_{ext}\times\Lambda} \frac{\left(V_{r,i}^{pred}(u,v,k) - V_i^{obs}(u,v,k)\right)^2}{\binom{|V_{ext}|}{2}}} \tag{5.7}$$

Using Equation 5.7, for each *Rank* value we compute the average of RMSE for all 20 runs and their 95% confidence intervals. Motivated by our results from the Subsection 5.5, we also integrate the Foci similarity with the NCN predictor. We take a two-fold sorting strategy by which we first sort prediction quadruples by using their NCN measures in a descending order. Next, we employ the social Foci similarity to sort all quadruples with the same NCN scores based on their social similarity in a descending order. We call this new predictor the *Foci-NCN* method. We show the TPR, FPR, precision, accuracy, and RMSE results for the NCN, Jacard, Min, Popularity, and Foci-NCN predictors in Figures 5.14, 5.15, 5.16, 5.17, and 5.18, respectively.

Figure 5.14 shows the TPR results for NCN, Jacard, Min, Foci-NCN, and Popularity methods. For small Rank values, the TPRs are low for all methods. We can also see that for small Rank values NCN and Foci-NCN perform better than the Jacard and Min methods; however, the difference between performances of methods based on the neighborhood similarity becomes negligible for large Rank values. Furthermore, the Popularity method always performs worse than other methods which is in match

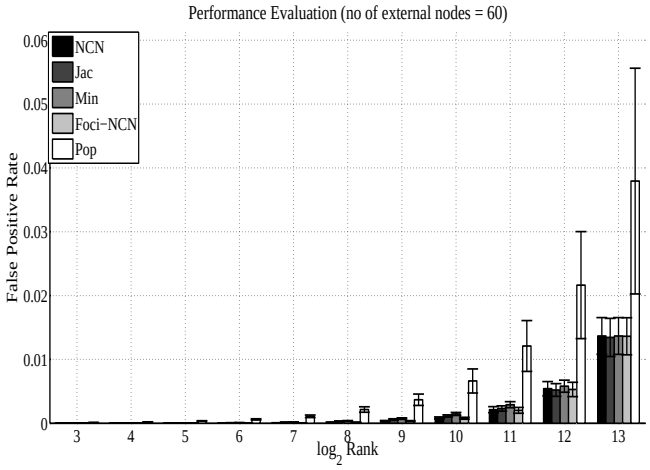


Figure 5.15: False positive rates of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06)

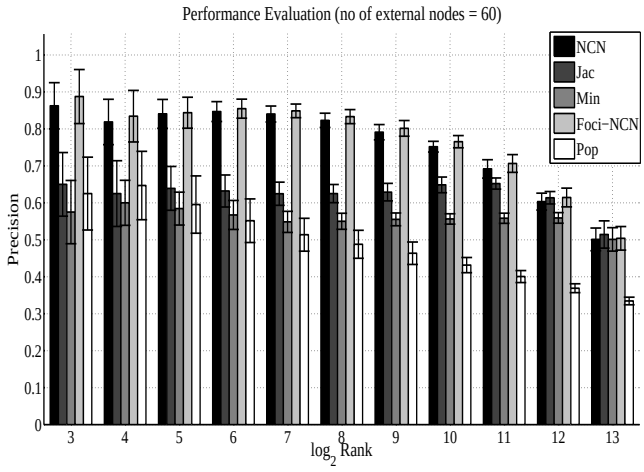


Figure 5.16: Precisions of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06)

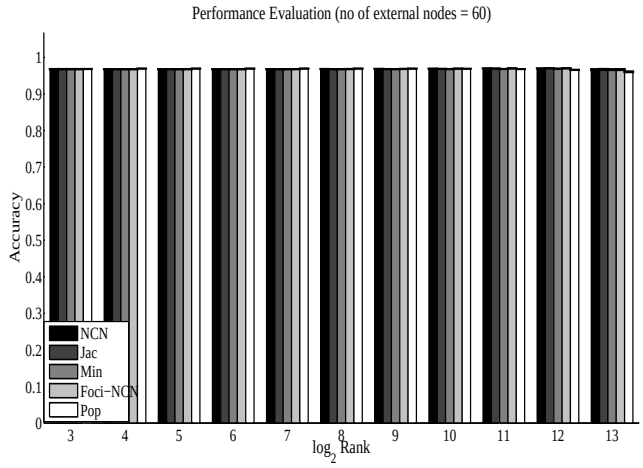


Figure 5.17: Accuracies of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06)

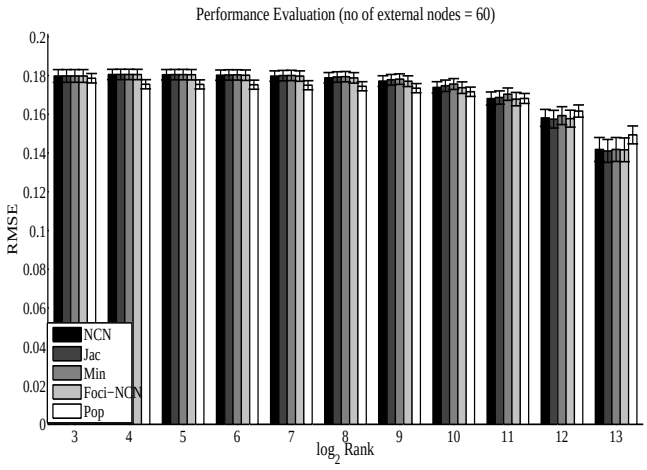


Figure 5.18: RMSE of NCN, Jacard, Min, Popularity, and Foci-NCN predictors (Infocom 06)

with our results from the Subsection 5.4. Figure 5.15 presents our FPR results for all five methods. Here, we can observe the same behavior as TPR results. Although the methods based on the neighborhood similarity perform very similarly for different Rank values, the Popularity method shows the worst FPR results.

Our precision results are presented in Figure 5.16. As we can see, combining the social information with NCN similarity in Foci-NCN method provides us slightly better precision results compared to other methods. This is in agreement with our results from the previous subsection. However, when we increase the Rank value to predict more missing contacts, the prediction methods based on the neighborhood similarity lose their predictive power and demonstrate similar performance. Furthermore, we observe that the Popularity method demonstrate the poorest precision results. This is again because unlike the methods based on the neighborhood similarity, the Popularity predictor does not contain any geographical information.

From Figure 5.17, we can see that the accuracy is always high. This is because contact graphs are usually sparse as was shown in Figure 5.11. As a result, even if the predictors do not infer any contacts between any pairs of external nodes, we still obtain a high accuracy. Therefore, the accuracy is not an interesting measure for the contact prediction task. Finally, Figure 5.18 shows the RMSE results for our five different predictors. When we increase the Rank number, we achieve smaller RMSE values. We also can observe that the Popularity method again demonstrates the worst RMSE results.

5.3.7 NCN Scalability

In previous sections, we showed that the NCN outperforms other predictors in different social settings. Our performance results were demonstrated for a particular ratio of external to total nodes (i.e. 75%). However, it is important to show how the predictors precision change over different values of external to total nodes ratio. Since the NCN predictions are statically more significant than other predictors, here we just show the scalability results of the NCN predictor for the Infocom 2006 and Rollernet datasets. Figures 5.19 and 5.20 demonstrate the scalability of the NCN predictor for the Infocom 2006 and Rollernet settings respectively. As we can see, the ratio of external nodes to total nodes have been changed from 20% to 80%.

There are several observations that one can make from the scalability results. First, in both settings the predictor precision drops quickly when the ratio is 20%.

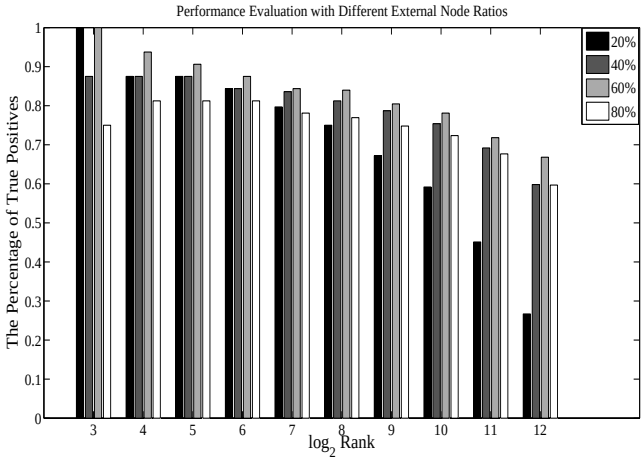


Figure 5.19: Scalability results for NCN predictor (Info06)

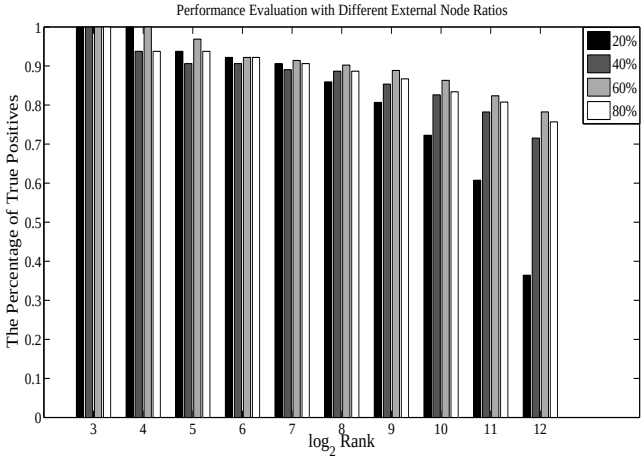


Figure 5.20: Scalability results for NCN predictor (Roller)

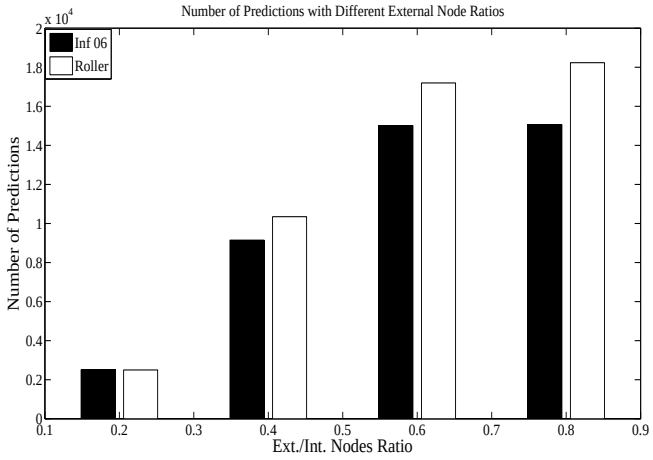


Figure 5.21: Number of predictions for NCN predictor (Infocom 2006/Roller)

This can be justified by considering the fact that in this case actually there are not many missing edges to predict. This becomes clear if we compute the number of predictions produced by the NCN method for different ratio values as shown in Figure 5.21. We see that when the number of external nodes is just 20% of the total number of nodes, the number of external pairs with an NCN greater than zero is significantly smaller than 4096 for both datasets. However, as we increase the ratio of external to total nodes, there are more missing edges that the NCN can predict. Finally, our scalability results show that the NCN predictor performs well when we increase the percentage of external nodes. In other words, when the number of external nodes increases, there are many missing edges in contact graphs that can be inferred by the NCN predictor successfully.

5.4 Why NCN Performs Well?

In Subsection 5.3.4, we saw that as we increase the number of predictions, the performances of our predictors drop. This is because when we increase the number of predictions, the similarity scores such as NCN drop. Figure 5.22 shows the relation between the probability of a contact between two external nodes' surrogates and their NCN value for different numbers of external nodes. As we see from the Figure 5.22, the NCN score and the contact probability are positively correlated. In other words, higher NCN scores imply higher contact probabilities. This explains why increasing the number of predictions reduces the performances of our predictors. According to

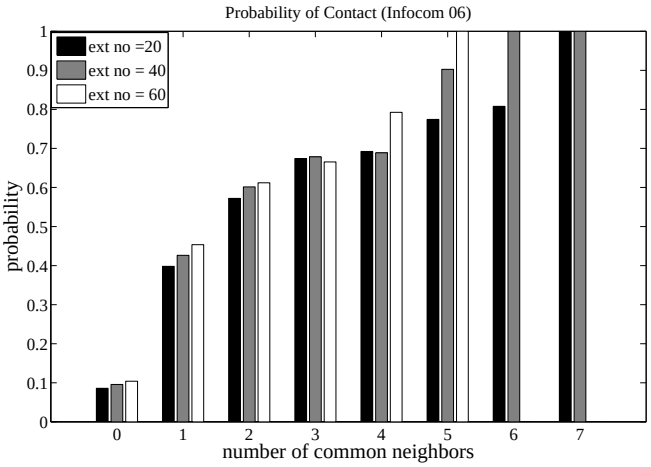


Figure 5.22: Probability of contact as a function of *NCN* for Infocom06 dataset

Figure 5.22, as soon as two external nodes' surrogates have even one node in common, their meeting chance increases by a factor of four which is quite significant. Figure 5.22 also shows that as the *NCN* increases, the performance of prediction also improves.

Our performance results clearly show that the number of common neighbors more accurately predicts missing contacts among external nodes than other predictors. In this section, we try to explain the performance of the number of common neighbors. As mentioned earlier, we expect spatial locality to increase the meeting probability between nodes. Thus, we need to estimate the distances between nodes by using a metric. Suppose two nodes u and v with similar radius range r are located at distance d from each other. If these two nodes have at least one common neighbor, i.e. a node is within their communication range, we can infer that the intersection area between their circles is greater than zero as shown in Figure 5.3. This requires an upper bound equal to $2 \times r$ for the distance between nodes u and v (i.e. $d(u, v) < 2 \times r$). If we assume a uniform distribution for nodes' locations on the plane, we can explain the success of the NCN.

Let us assume that we have n nodes with radius range r which are uniformly distributed in a region with an area A (e.g. a conference room). In this graph, there is an edge between two nodes u and v if their distance is less than r (i.e. $d(u, v) < r$). The resulting graph is a random geometric graph (RGG) [71]. Researchers have used RGGs to model the wireless ad-hoc networks [32]. A direct consequence of the dependency of links on the geometric distance between nodes is

that there is an increased probability of two nodes to be connected with each other if they have a common neighbor. We also observed this increasing pattern in our contact probability graph (Figure 5.22). Let E_i denote the event of occurrence of edge e_i in the described RGG. The probability of occurrence of an edge e_i between a pair of nodes is $Pr(E_i) = \frac{\pi r^2}{|A|}$. For two distinct edges $e_i = (u, v)$ and $e_j = (u, w)$, we have $Pr(E_iE_j) = Pr(E_i)Pr(E_j) = (\frac{\pi r^2}{|A|})^2$. For three distinct edges $e_i = (u, v)$, $e_j = (u, w)$, and $e_k = (v, w)$, authors of [85] derived the probability that these three edges form a triangle as follows:

$$Pr(E_iE_jE_k) = \frac{(\pi - \frac{3\sqrt{3}}{4})\pi r^4}{|A|^2} \tag{5.8}$$

Using Equation 5.8, we derive the probability for occurrence of edge e_i between nodes u and v given they have one common neighbor (node w) as follows:

$$Pr(E_i|E_jE_k) = \frac{Pr(E_iE_jE_k)}{Pr(E_jE_k)} = \frac{\pi - \frac{3\sqrt{3}}{4}}{\pi} \tag{5.9}$$

Equation 5.9 gives $Pr(E_i|E_jE_k) \approx 0.58$ which is obviously more significant than the small value of $Pr(E_i) = \frac{\pi r^2}{|A|}$ ($|A| \gg \pi r^2$) which again justifies our observation in Figure 5.22 where the probability of contact suddenly jumps as soon as nodes have one common neighbor. Unfortunately computing the link probability when the number of common neighbors is greater than one becomes much harder. This is mainly because of the dependency issues that the third and fourth edges add to the conditional probability formulation.

5.5 Discussion

In this chapter, we have studied the problem of contact prediction in the context of mobile social networks. We have described different methods for predicting missing contacts. We have examined our methods by using real contact traces collected from different social settings. Our results show that the time-spatial based scores provide the most reliable results for predicting missing contacts among external nodes. We have also studied the power of social profiles to predict human mobility. We

have shown that combining social information with time-spatial information provides better performance results than using each of them independently. We believe that our contributions have significant practical values because they allow researchers to study properties of large scale contact graphs by sampling only a portion of the original graphs. Our results are also important for mobility modeling since they explain how people contact each others in different social settings such as conference and campus environments.

It is crucial to highlight the applicability of using our proposed contact predictors for other large datasets collected from other social settings which were not examined in this chapter. The generality of our methods can be justified by considering our effort in this chapter for testing the proposed predictors by using a variety of contact data collected from two conferences, two large campus environments, and one outdoor event. We also need to emphasize that our predictors owe their success to the time-spatial locality property which play a significant role in the structure of contact graphs. This property is independent from the social setting for which we have contact data.

Chapter 6

Predicting Human Contacts using Supervised Learning

In this chapter, we study the same problem as the previous chapter; however, we explore the possibility and benefits of using supervised learning algorithms for predicting missing contacts in existing contact traces. For predicting hidden contacts, we employ a supervised learning approach in which we use training data to devise a classifier function for predicting the classes of unseen data. First, we extract several features by using information from the underlying structure of contact graphs, social profiles of people, and static sensors. We use two supervised learning classifiers namely *Logistic Regression* and *K-Nearest Neighbor* for predicting hidden contacts. We validate our classifiers by taking two different approaches as described in Section 6.2.4. Finally, we examine the effect of a node *Degree Centrality* that is the total number of contacts a node had during a social event on its contact predictability level. The results of this chapter was published in Simplifying Complex Networks workshop in 2012 [45]. The contributions of this chapter are as follows:

1. We show the applicability of using supervised learning algorithms by taking two different approaches for validating our classifiers.

2. We demonstrate that the *number of common neighbors* and the *total overlap time* are the most significant features in contact prediction.

3. Finally, we show that contacts of nodes with high centrality are more predictable than nodes with low centrality.

6.1 Related Work

As we have mentioned in the Chapter 2, Nowell et al. studied the link prediction problem in a citation network where they extracted several features for predicting future collaborations among researchers [57]. Hasan et al. extended the Nowell's work by employing several supervised learning classifiers including Support Vector Machine (SVM) with two different kernels, Decision Tree, Multilayer Perception, K-Nearest Neighbor (KNN), Naive Bayes, RBF Network, and Bagging for predicting future co-authorship in citation networks [34]. The authors extracted non-topological features in addition to topological features where the extracted features were proposed to measure the proximity of nodes in citation networks. Their prediction results show that supervised learning is an efficient way to formulate the link prediction problem in social networks. Their results also show that SVM classifier with Radial Basis Function (RBF) kernel provides the best performance; however, other classifiers such as Decision Tree, KNN, and Multilayer Perception also demonstrate similar performance.

Leskovec et al. also used the machine learning framework to predict the sign of links in social networks where positive links may indicate friendship relations and negative links may indicate opposite relations [55]. They extracted several features and fed them into a Logistic Regression classifier in order to predict signs of links in social networks. Finally, authors of [25] proposed a set of graph-based features for link prediction in social networks. They used a variety of popular machine learning methods such as C4.5 Decision Trees, KNN, SVM and so on.

6.2 Problem Definition

Our problem description is quite similar to the problem that we tackled in Chapter 5. We again denote the set of sensors with V_{int} where sensors are considered as *internal nodes*. We also designate the set of *external nodes* that are nodes which do not carry any sensors with V_{ext}. We divide the experiment time into equal intervals of τ seconds called *time intervals* as the previous chapter. We choose $\tau = c \times T$ where c is a constant integer, and T is the *inquiry interval* of wireless sensors, namely the time gap between two consecutive sensings. The coefficient c is usually chosen to be one or two.

Let $\Lambda_k = [t_0 + k\tau, t_0 + (k+1)\tau]$ denote the k^{th} time interval where $0 \leq k < k_{max}$

and t_0 is the starting time of the experiment. We show people's interactions during the k^{th} time interval with an undirected contact graph G_k that contains contacts between people in Λ_k. In $G_k = (V_k, E_k)$, edges in $V_{int} \times (V_{int} \cup V_{ext})$ are known while edges in $V_{ext} \times V_{ext}$ are missing. Our objective is to predict these missing edges by employing a binary classifier.

6.3 Contact Prediction using Classification Algorithms

In this section, we review two supervised classification algorithms by which we formulate the relationship between a dependent variable (i.e. output variable) and one or more independent variables (i.e. features). In our case, we use these classifiers to estimate the probability of a contact between a pair of external nodes as a function of their feature vector.

6.3.1 Logistic Regression Overview

There are many problems where we want to find the class to which an item belongs. In our problem, each pair of external nodes such as (u, v) can belong either to an *edge* or to a *non-edge* class. Thus, we want to find the probability that a given pair of external nodes belongs to *edge* class that is the probability that a contact actually happened between them. This can be formulated as a binary classification problem where the output variable $y \in \{0, 1\}$. In particular, logistic regression can be used to formulate our problem where the hypothesis function satisfies the $h_\Theta(x) \in [0, 1]$ condition. In particular, we choose the hypothesis function as below [9]:

$$h_\Theta(x) = g(\Theta^T X) = \frac{1}{1 + e^{-\Theta^T X}}, \tag{6.1}$$

where $g(.)$ is the logistic function, X is the feature vector, and Θ is the parameter of the model that we want to learn. If we show the *edge* class with one and the *non-edge* class with zero, we can compute the contact probability between u and v using the logistic regression as follows:

$$p(y = 1|X; \Theta) = h_\Theta(x) = 1 - p(y = 0|X; \Theta) \tag{6.2}$$

We compute the likelihood of the parameter Θ as follows:

$$L(\Theta) = p(\vec{y}|X;\Theta) = \prod_{i=1}^{m} p(y^{(i)}|X^{(i)};\Theta), \tag{6.3}$$

where m is the size of training set. We find the parameter Θ such that it maximizes the likelihood $L(\Theta)$.

6.3.2 K-Nearest Neighbor Overview

Another method that we use for classification is k-nearest neighbor method (i.e. KNN). We employ this method to estimate the probability distribution of edge existence between external node pairs given their feature vectors. KNN is a non-parametric estimator where it does not make any assumptions about the probability distribution function. However, logistic regression makes specific assumption about the form of the logistic function. In KNN, we have all of our training points in a d-dimensional space where d is the number of features. When we want to find out the label of a given external pair such as (u, v), we first find the K nearest neighbors of (u, v) in the feature space using the Euclidean distance. Then, we classify (u, v) by returning the class that majority of its K nearest neighbors belong to [9].

6.3.3 Features Extraction

While configuring a supervised learning classifier, we must explore all important features that may have influence on the output variable that we wish to predict. Here, we need to explore all possible independent variables that have impact on the probability of contacts between two external nodes in the time interval Λ_k.

Contact Graph-based Features

First, we focus on features that are based on local properties of contact graphs. In particular, for a given time interval Λ_k we construct the partial contact graph G_k. Next, we extract several features from the structure of G_k in order to predict the probability of contacts. For $G_k = (V_k, E_k)$, let $N^k(u)$ denote the neighborhood set of node u that contains all nodes which had at least one contact with node u in Λ_k:

$$N^k(u) = \{v|(u, v) \in E_k\} \tag{6.4}$$

Using the neighborhood set of node u, we devise several degree based features.

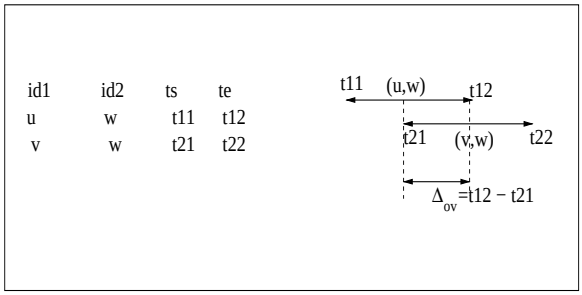


Figure 6.1: Contact duration as a feature.

First, we see that degree of a node $u \in V_k$ (i.e. $|N^k(u)|$) represents the number of contacts that node u had during Λ_k. We assume that if node u has a high number of contacts, it is more likely to contact a randomly chosen node than a node v with low number of contacts. As a result, for a given pair of external nodes such as (u, v) we use $|N^k(u)|$ and $|N^k(v)|$ as the first two degree-based features. Moreover, for a given pair of nodes such as (u, v) we assume that the contact probability between them not only depends on their individual degrees, but it also depends on the product of their degrees in Λ_k. Therefore, we choose $|N^k(u)| \times |N^k(v)|$ as the third feature.

If two nodes u and v are in close proximity of each others, they are likely to contact each other in the near future. In previous chapters, we showed that the number of common neighbors between a pair of nodes can be employed to estimate the geographical distance between them [49]. We similarly use the number of common neighbors between two nodes as the fourth feature that is $ncn(u, v) = |N^k(u) \cap N^k(v)|$.

Contact Duration

Contact duration is another important feature that is useful for the prediction task in addition to number of contacts. As it is shown in Figure 6.1, not only the fact that two nodes such as u and v have seen the same node w influences the probability of a contact between u and v, but the overlap time that node w has been in contact with u and v also matters (i.e. $\Delta_{ov}(u, v; w) = t_{12} - t_{21}$). As a result, we define the total overlap time feature as shown in Equation 6.5 where for each pair of external nodes we compute the total overlap time that these two nodes have spent with the same third nodes in Λ_k.

$$T^k_{ov}(u, v) = \sum_{w \in (N^k(u) \cap N^k(v))} \Delta^k_{ov}(u, v; w) \tag{6.5}$$

Social Information

The third class of features that we use in our feature vector is the social similarity between nodes. Our main intuition is that mobile nodes are more likely to contact other nodes that are socially similar to them. As it was mentioned in Chapter 3, in the Infocom 2006's data, participants reported a brief version of their social profiles including their affiliations, research interests, country of birth and so on. We use Equation 4.2 presented in Chapter 4 to compute the Foci similarity between nodes as another feature.

Static Sensors

The Infocom 2006 dataset includes information from static nodes which were deployed in different conference rooms to detect mobile nodes in that room. These static nodes have longer radio ranges than mobile sensors. We use the data from static nodes as another feature for our learning algorithm. We add a new feature which just counts the number of common static nodes which were seen by a pair of external nodes in Λ_k. This feature basically tells us if the corresponding mobile nodes are in the same room or not.

6.3.4 Training/Validating Binary Classifiers

Since in human contact traces we do not have any information about the contacts between external nodes, there is not any way for us to validate the predicted contacts between them. To get around this issue, we take the same approach as Chapter 5 by choosing a random subset of internal nodes and labelling them as *external surrogates* (i.e. V_{surext}). These external surrogates play the same role as external nodes. We remove all contacts observed by external surrogates that are all edges such as $(u, v) \in V_{surext} \times V_{surext}$ [49]. We generate the partial contact graphs by removing edges among external surrogates. We use these partial contact graphs to train and test our classifiers. Next, we describe two approaches for training and testing our classifiers.

Approach I

In the first approach, we test the possibility of using a contact dataset such as A as the training data in order to predict the missing contacts for another contact dataset B. Using the Infocom 2005, Infocom 2006, and Rollernet datasets we examine how

Table 6.1: Approach I performance results (Logistic Regression/KNN)

Training Data	Info 05	Roller	Roller
Test Data	Info 06	Info 05	Info 06
TPR	0.29/0.31	0.39/0.38	0.35/0.41
FPR	0.05/0.11	0.08/0.11	0.07/0.09
Correctly classified	79%/75%	75%/72%	80%/77%
RMSE	0.44/0.42	0.48/0.45	0.44/0.40

accurately we do the prediction if for instance we use the Infocom 2005 as the training data while using the Infocom 2006 as the test data.

Approach II

In the second approach, we use the well known *k-fold cross validation* technique in which we use part of the dataset as the test data while the rest of data is used as the training data. This way we train our predictor using the training data and then we use the test data to evaluate the learned algorithm.

6.4 Prediction Results

In this section, we present our prediction results using logistic regression and KNN classifiers. We use the Weka software for testing the chosen classifiers [4]. For training and validating our classifiers, we use the two approaches described in the previous section. For our contact datasets, we use the Infocom 2005 and Infocom 2006 datasets that were collected from Infocom conferences in 2005 and 2006, respectively [39]. We also use the Rollernet dataset (i.e. Roller) containing the contacts from a set of people who participated in a rollerblading tour in Paris [78]. Readers can find the detail descriptions of these datasets in Chapter 5.

In both Infocom 2005 and 2006, we use the data collected on the first day of the conference. We choose the time interval (τ) to be 240, 240, and 30 seconds for the Infocom 2005, 2006, and Rollernet datasets, respectively. For logistic regression, we use 0.5 as our threshold where we classify each pair of external surrogates with a predicted probability greater than the threshold as an *edge*. For KNN, we choose $K = 3$. In all of our experiments, we label 60% of nodes as external surrogates. We repeat each experiment ten times with different subsets of external surrogates and show the averages.

Table 6.2: Approach II performance results (Logistic Regression/KNN)

Session Type	Keynote	Lunch	Coffee
TPR	0.18/0.24	0.37/0.40	0.41/0.43
FPR	0.03/0.08	0.04/0.07	0.02/0.02
Correctly classified	81%/78%	84%/81%	92%/92%
RMSE	0.42/0.40	0.39/0.36	0.26/0.24

6.4.1 Approach I Results

In approach I, we use the Infocom 2005, 2006, and Rollernet datasets. We choose $k = 81$, $k = 15$, and $k = 32$ for the Infocom 2005, 2006, and Rollernet respectively. Table 6.1 shows the performance results for logistic regression and KNN classifiers. As we can see, both classifiers outperform a random predictor. Thus, we can conclude that our three datasets have similar structures such that we can train our classifiers by using one of them while using the other one as the test data. Moreover, we observe that using the Rollernet as the training data provides better prediction results than using the Infocom 2005.

6.4.2 Approach II Results

In a conference setting, different events happen during each day such as keynote talks, panels, coffee/lunch breaks, and regular sessions. Here, we just focus on three different types of sessions including keynote, lunch, and the last coffee break. In our second approach, we use the *k-fold cross validation* method to evaluate our predictors. We use the Infocom 2006 dataset. We partition the V_{surext}-induced subgraph of CG_k into five different subsets. We repeat our experiment five times where each time we use one subset of the induced subgraph as the test data and the rest as the training data. Finally, we compute the overall performance by computing the average of all trials. Our 5-fold cross validation results for logistic regression and KNN classifiers are shown in Table 6.2.

As we can see, the true positive rate (i.e. TPR) is low. This is because there is not enough information in the extracted features. Moreover, we found several instances of external pairs where there were not any internal nodes around them. In this case, there is not much information for our classifiers to use for prediction. Even if TPR is not high, we see that the false positive rate (i.e. FPR) is low.

Table 6.3: The average rank for different features (Infocom 2006)

Session Type	Keynote	Lunch	Coffee
degree	4	5	5
degree	7	7	7
degree product	3	3	6
ncn	1	1	2
total overlap	2	2	1
social	5	6	4
ncsn	6	4	3

6.4.3 Features Significance

In this part we compare the significance of different features described in Subsection 6.2.3. This is important because we can find out what features play important role in contact prediction. We use the Infocom 2006 as our dataset and choose 60% of nodes as external surrogates. We extract the feature vectors for all possible pairs of external surrogates in our test data. We use this test data to rank different features. It is important to note that for each type of session we repeat our experiment ten times where in each trial we randomly pick 60% of internal nodes and label them as external surrogates. For ranking the features, we use three different algorithms including information gain, gain ratio, and Chi-Square. We show the average ranks of different features for the three types of sessions in Table 6.3. As we can see, the number of common neighbors and the total overlap time are the most significant features in all sessions. This is because both of these features contain the geographical proximity data. The product of degrees and the number of common static nodes also appear as the next important features.

We can also evaluate the importance of our features for different types of sessions by using their class density distributions. We show the class density distributions of the four most important features for lunch break in Figure 6.2. One interesting pattern that we have found in the class density distribution is that *edge* and *non-edge* classes become distinguishable when the number of common neighbors and the total overlap time increase. We observe almost the same pattern for degree product and social similarity but these later features are not as significant as the former ones.

We observed that the number of common neighbors is the most significant feature in contact prediction task. Now, we would like to compare the performance results of our previous supervised classifiers with a linear classifier that only uses the number

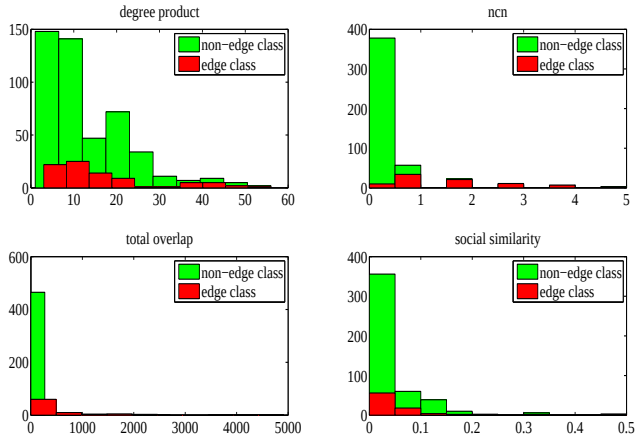


Figure 6.2: Class density distributions for lunch session (Infocom 2006)

Table 6.4: Performance results of NCN linear classifier (Infocom 2006)

Session Type	Keynote	Lunch	Coffee
TPR	0.22	0.39	0.29
FPR	0.05	0.04	0.01
Correctly classified	80%	84%	92%

of common neighbors for the prediction task. We designate a threshold for *ncn* and classify all pairs that have an *ncn* value greater than the threshold in the *edge* class and the rest of pairs in the *non-edge* class. We choose the threshold to be two. The results are shown in Table 6.4. Comparing Table 6.4 with 6.2, we observe that using all features do not give us a significant improvement over the linear classifier that uses only the ncn feature. This again shows the significant role of the ncn feature in predicting the missing contacts.

6.4.4 Centrality Effect on Predictability

One interesting question that can be asked is what nodes are more predictable. We would like to select external nodes such that their contacts become more predictable. In particular, we want to study the effect of degree centrality of nodes on their predictability. Let us define the degree centrality of node u in time interval Λ to be the total number of contacts that node u has had during Λ. We use the dataset of Infocom 2006 and focus on the first day of the main conference. Assuming that we have the full information of all nodes, we compute the centralities of all internal

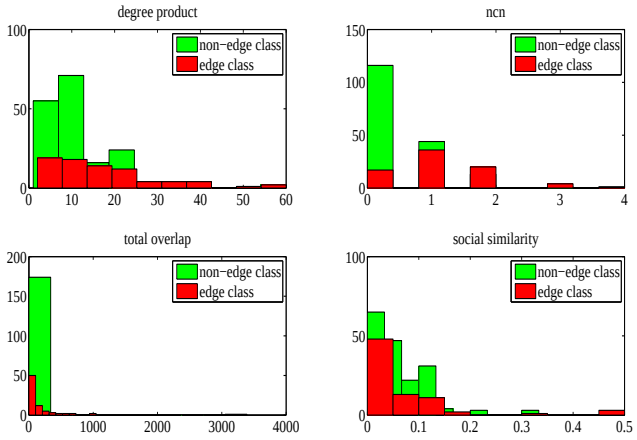


Figure 6.3: Class density distribution of external nodes with highest centrality (Infocom 2006: keynote)

Table 6.5: The effect of centrality on predictability

External Type	Most Cent.	Least Cent.
TPR	37%	0%
FPR	15%	0%
Accuracy	70%	92%

nodes using the entire data of the 9-hour period. We sort all nodes according to their centralities in a descending order.

For our experiment, we first choose the top 30% of nodes from the sorted list as the external nodes with the highest centralities. Secondly, we choose the bottom 30% of nodes from the sorted list as the least central external nodes. For prediction purpose, we just focus on the keynote session. Figures 6.3 and 6.4 show the class density distributions when external surrogates are chosen to be the most and the least central nodes, respectively. We can see that choosing external surrogates from the most central nodes makes the class density distribution of *edges* to be more distinctive form *non-edges* than when we choose the least central nodes as external surrogates.

Thus, we expect that nodes with high centrality to be more predictable than nodes with low centrality. The reader should note that if one chooses the most central nodes as external nodes, then she would have a higher number of pairs that fall in the *edge* class which in return helps the classifier for better prediction results. We use KNN with $K = 3$ with 10-fold cross validation to see how accurately we predict if we

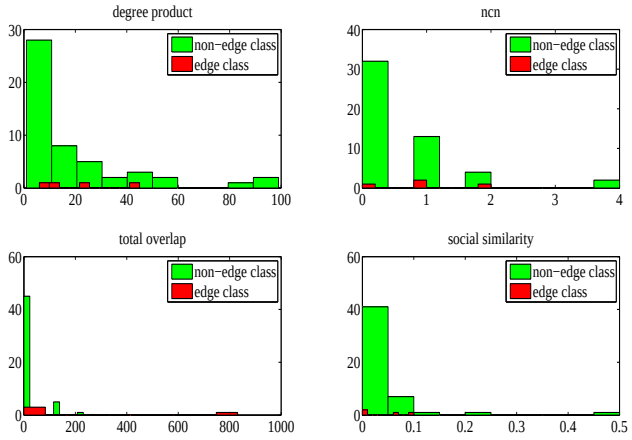


Figure 6.4: Class density distribution of external nodes with lowest centrality (Infocom 2006: keynote)

choose external nodes differently. The performance results of two approaches are shown in Table 6.5. We observe that choosing nodes with less centrality as sensors helps classifiers achieve better prediction results. These results are important because they illuminate how practitioners should choose the sensor nodes for sampling human contacts in order to achieve better predictions.

6.5 Discussion

In this chapter, we employed logistic regression and KNN classifiers from the machine learning area for predicting missing contacts. We did this by extracting a set of different features. Interestingly, we demonstrated that it is possible to use a contact dataset A as the training data in order to predict the missing contacts of a different dataset B. We also showed that the number of common neighbors and the total overlap time play the most significant roles in predicting human contacts. Finally, we showed that nodes with high centrality provide better prediction results than nodes with low centrality.

Chapter 7

Fast Information Spreading in Social Networks

Consider a scenario in which we have a piece of information such as an image or a video on our cellphone. Our goal is to send this piece of information to everyone in the network in the shortest amount of time. However, in each time step we are restricted to choose one of our friends from our contact list and send the message to her. Everybody else also has to follow the same algorithm in order to spread the message. Now, our research question is to determine in each step who is the best person to contact and send the message to in order to spread the message to everyone quickly. In this problem, known as information spreading, we assume that when a person receives a message from one of her neighbors in the social network, she immediately accepts the message and stores it on her device for further spreading. This assumption makes our problem different from diffusion of an innovation in a social network. However, the nature of our problem still remains similar to the diffusion of innovation process because the position of a node in the network as well as the underlying structure of the social network play essential roles in how fast information spreads.

Fast information spreading has several applications such as advertising a piece of message to everyone through social contacts in a social network. Here, we first analyze the running times of already proposed information spreading algorithms in social networks to find the fastest one. In particular, we study three information spreading algorithms including the *random push-pull* [64], an algorithm proposed by Censor et al. [12], and another algorithm proposed by Doerr et al. [21]. For the rest of this chapter, we refer to the algorithms proposed by Censor et al. and Doerr et al.

as the *Censor* and *Doerr*, respectively. For our analysis, we assume a synchronized communication model where initially each node has a unique message.

We first compare the running times of the three information spreading algorithms empirically on a social graph collected from the Facebook website [80]. Interestingly, our empirical analysis shows that the Censor algorithm outperforms both the random push-pull and the Doerr algorithms. We justify our performance results by showing that the Facebook graph has a *core-periphery* structure which cannot be explained by random or Preferential Attachment models. Although the core of the Facebook graph has an expanding structure, its periphery consists of a large number of small communities (i.e. *1-whiskers*) that are weakly connected to the core. Furthermore, we mathematically prove that these 1-whiskers act as communication bottlenecks for information spreading.

The *betweenness centrality* is a well-known measure in social networks by which we can compare the importance of nodes and edges from a structural point of view [26]. In the last part of this chapter, we employ the betweenness centrality measure to identify bottleneck edges and visit them with a higher probability. Our results show that we can spread information in social networks quickly if we identify the bottleneck edges by using the centrality measure. The results of this chapter was partly published in the third IEEE International Conference on Social Computing [44].

7.1 Background and Related Work

Let us represent the underlying structure of a social network with graph $G = (V, E)$. A *cut* is a partition of V into two disjoint subsets. For a given cut $(S, V \setminus S)$, we define the *conductance* φ of the set S as follows:

$$\varphi(S, V) = \frac{\sum_{i \in S, j \in V \setminus S} P_{ij}}{|S|}, \tag{7.1}$$

where P is the stochastic matrix associated with G [12]. The conductance of graph G is then defined to be:

$$\Phi(G) = min_{S \subseteq V, |S| \leq \frac{n}{2}} \varphi(S, V) \tag{7.2}$$

In the random push-pull algorithm, in each round every node chooses one of

its neighbors randomly to exchange its messages with. Mosk-Ayoma and Shah [64] showed that for a $\delta \in [0,1]$, the random push-pull algorithm terminates in $O(\frac{\log n + \log \delta^{-1}}{\Phi(G)})$, with probability of at least $1 - \delta^{-1}$, where $|V| = n$ and $\Phi(G)$ is the conductance of G as defined in Equation 7.2. Conductance of a graph (i.e. $\Phi(G) \in [0,1]$) determines how well-connected the graph is. While well-connected graphs (e.g. cliques) have large conductance values, the graphs that have many communication bottlenecks (e.g. paths) have low conductances. Here, the main obstacle to speeding up the information spreading is the size of graph conductance. Thus, the random push-pull performs fast for those communication graphs which have large conductance. However, if a given graph has a low conductance (e.g. $\Phi(G) = O(\frac{1}{n})$), the number of rounds needed to spread messages to all nodes is a polynomial function of n.

Censor et al. identified a class of graphs that have low conductance but large weak conductance [12]. While conductance measures the connectivity of the whole graph G, weak conductance of a graph G (i.e. $\Phi_c(G)$) measures the *best* connectivity among subsets that include each node. The size of subsets depends on a c value. We refer the reader to [12] for the formal definition of the weak conductance. One example for a graph with low conductance but large weak conductance is a path of c cliques where each of them has $\frac{n}{c}$ nodes. While the path of c cliques has low conductance (i.e. $\Phi(G) = O(\frac{1}{n})$), each clique is a very well-connected component (i.e. $\Phi_c(G) = O(1)$).

To speed up the running time of spreading algorithms for graphs with many communication bottlenecks, one needs to identify them. Censor et al. used a hybrid approach to identify the bottlenecks. In their algorithm, every node has a circular list called the *bottleneck list* which initially contains all of its neighbors. In even rounds each node chooses a random neighbor to contact, but in odd rounds each node chooses the next node to be contacted from the top of its bottleneck list. In the Censor algorithm, each node v permanently keeps one of its neighbors such as u in its bottleneck list if v contacts u at round r and receives u's message (i.e. $m(u)$) for the first time directly from u. If v receives $m(u)$ in any other ways, v removes u from its bottleneck list. If v receives $m(u)$ for the first time from node u itself when v contacts u directly, this implies that node v's community is probably weakly connected to node u's community; otherwise, it could have received $m(u)$ earlier from one of its other neighbors. Censor et al. proved that their algorithm improves the running time from polynomial to poly-logarithmic time for graphs with low conductance but large weak conductance [12].

For mathematical analysis of the Censor algorithm, we associate colors with the

directed version of our social network in that every node has a directed edge to its neighbors. Initially, all edges have *Grey* colors specifying that all nodes have listed all their neighbors in their bottleneck lists. Whenever a node such as v for the first time contacts a node u and receives $m(u)$ directly from u, the edge (v, u) turns to *Black* color meaning that v will not remove u from its bottleneck list anymore. Finally, if v in round r has to remove u from its bottleneck list, the edge (v, u) turns to *White* color. The reader should note that this color illustration is just for analysis purposes [12].

Chierichetti et al. proved that for a graph with conductance of Φ, $\hat{O}(\Phi^{-1} \log n)$ number of rounds is needed to broadcast a message from a source node to all other nodes in the graph using the random push-pull where $\hat{O}(.)$ notation hides a $polylog(\Phi^{-1})$ factor [17]. Chierichetti et al. also analyzed the running time of rumor spreading where a message has to be delivered to all other nodes in a network with preferential attachment model [16]. They proved that the random push-pull strategy takes $O(\log^2 n)$ rounds to terminate. Recently, Doerr et al. improved Chierichetti's results for preferential attachment graphs [21]. They showed that the random push-pull spreads a message starting from a node to all other nodes in the graph within $\Theta(\log n)$ rounds with a high probability if the graph has power-law node degree [21].

Furthermore, Doerr et al. proposed a new algorithm that works slightly different from the random push-pull in that in every round each node contacts one of its neighbors uniformly at random except the one that it contacted in the previous round [21]. They proved that their new algorithm requires $\Theta(\frac{\log n}{\log \log n})$ rounds to spread a message to all nodes in networks with power-law node degree. However, our empirical and mathematical analysis show that the Censor algorithm beats the other two spreading algorithms.

Freeman formally formulated three kinds of centralities in social networks as three different ways for comparing nodes' importance in a network from a structural point of view [26]. The first measure is the *Degree Centrality* which just takes into account the degree of a node and is useful if we have concerns with communication activity of nodes. The second one is the *Closeness Centrality* which measures how close a node is to other nodes in the network. Finally, the last one which is most relevant to our work is the *Betweenness Centrality*. The betweenness centrality of a node in graph G is proportional to the number of shortest paths between all pairs of nodes that go though that specific node. Node betweenness centrality can be expanded to edges where an edge with a high betweenness centrality occurs on a large number of shortest

paths between pairs of nodes in a network. It is also important to mention that edges with high betweenness centralities play critical roles in the graph as they occupy a position at the interface between tightly-knit clusters. In fact, several researchers have used the edge-betweenness centrality in order to find communities in social networks by removing edges with the highest betweenness [68].

Granovetter discovered that most of times job seekers find their new jobs through their acquaintances instead of their close friends [31]. To explain this he identified two different types of links (or ties) in social networks: strong links and weak links. While strong ties connect us to our close friends from our community, weak ties connect us to our acquaintances who belong to other communities. As a result, it is more likely to find a job through our weak ties instead of strong ties because our acquaintances have access to information that we otherwise do not have access to. In other words, weak ties act as bridges by connecting us to parts of the network which would be otherwise far away from us.

In [70], authors found that there is a negative correlation between links' betweenness centrality and their strengths in mobile communication networks. They simulated how fast a message diffuses in a social network by using an epidemic model. Their results highlight the importance of weak ties for speeding up information spreading. Finally, in [7] authors analyzed how information spreads in the Facebook website. They found that although stronger ties are more socially influential, weak ties are responsible for the spreading of novel information. Thus, their results highlight the importance of weak ties for information spreading in online social networks such as Facebook. This is our main motivation for exploiting the edge-betweenness centrality in social networks in order to speed up information spreading.

7.2 Problem Definition

Information Spreading: For a given communication graph $G = (V, E)$, suppose every node $u \in V$ has a unique message $m(u)$ that is identified by its node *id*. In an information spreading algorithm, every node u chooses one of its neighbors (e.g. $v \in N(u)$) in each round r according to its algorithm and exchanges its current messages with the selected neighbor. The algorithm terminates when all nodes receive all other nodes' messages (i.e. by the time when every node has n unique messages in its queue where $|V| = n$). We list the computation and communication constraints that every node has to obey as follows:

- The only information that is known to each node is the set of its neighbors and the number of nodes in the graph (i.e. $|V| = n$). In other words, the remaining structure of the graph is unknown to all nodes.
- The communication is synchronous where in each round r every node u contacts one of its neighbors to exchange its messages with (i.e. push-pull model).
- In each round, every node contacts *one* of its neighbors to exchange its messages with that neighbor. However, in each round a node can be contacted by several of its neighbors and exchanges its message with all of those nodes.

We assume that in each round, the required computation for node u to exchange its message with other nodes (i.e. including the one that u contacts and the neighbors that contact u by themselves) as well as other required internal computations for u take constant number of steps. Therefore, the time cost of the spreading algorithm is the number of required rounds until all nodes receive all messages of the network.

7.3 Empirical Analysis

In this section, we analyze the running times of the random push-pull, Censor, and Doerr algorithms by using real data. We also employ an efficient framework to study the community structure in social networks. Our observations from this section helps us in the mathematical analysis of running times of information spreading algorithms.

7.3.1 Real Data Description

In this chapter, we use the social network (i.e. $G = (V, E)$) of *New Orlean* regional network in the Facebook collected by Viswanath et al. [80]. Our total dataset has 63731 nodes and 817090 edges. Since the original graph is not connected, for the rest of our analysis we focus on the largest component of the original dataset that has $|V| = 63392$ nodes and $|E| = 816886$ edges.

7.3.2 Empirical Analysis of Information Spreading Algorithms

For the first part of our analysis, we run the random push-pull, Censor, and Doerr algorithms by using the Facebook data in order to compare their running times. We assume that each node initially has one message identified by its node id. We run

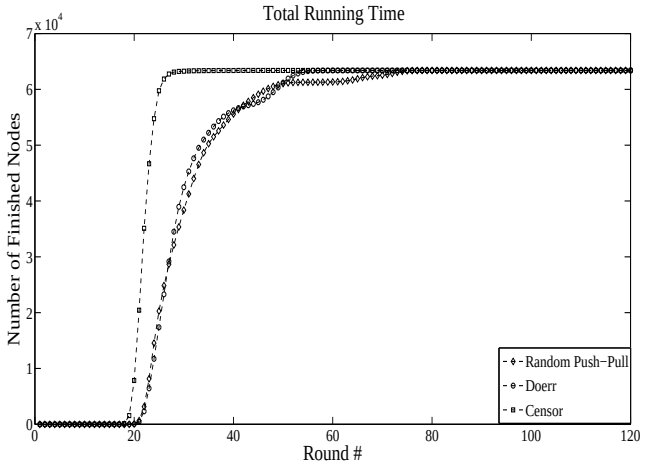


Figure 7.1: Running times of the random push-pull, Doerr, and Censor algorithms in the Facebook graph

each spreading algorithms forty times and compute their averages. The results for the algorithms' progress versus round number are shown in Figure 7.1. As we can see, the running time of the Censor algorithm is better than both random push-pull and Doerr strategies.

We can make several interesting observations from Figure 7.1. First, we find that 95% of nodes terminate in 25, 43, and 57 number of rounds in Censor, Doerr, and random push-pull algorithms respectively. This shows that in Censor algorithm 95% of nodes receive all messages at least twice faster than the random push-pull. Moreover, we can see that after a few rounds, the number of nodes that terminate start growing exponentially as a function of round number. This is because the underlying communication graph has an expanding structure. Our graph analysis result shows that our social network has a core-periphery structure where the core of the network has an expanding structure, which we describe in the next subsection.

7.3.3 Finding Bridges as Communication Bottlenecks

As the speed of information spreading in any graph is influenced by its underlying structure, identifying the community (or cluster) structure of the given graph allows us to better understand the running times of information spreading algorithms. Communities are considered as subsets of nodes that are internally well connected; however, they are loosely connected to the rest of network. This informal definition for communities is related to the conductance definition where a set S is considered

Figure 7.2: A sample of detected 1-whiskers in Facebook graph

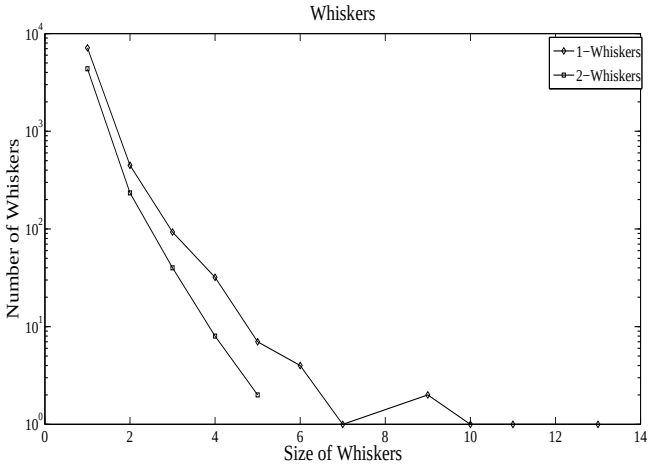


Figure 7.3: The number of 1-whiskers and 2-whiskers as a function of their sizes

as a good community if we can cut it easily from the rest of graph G (i.e. S has a low conductance). Finding communities play an important role in information spreading as nodes from the same community receive each others messages faster than messages of nodes from different communities.

For identifying communities inside the Facebook graph, we employ the Tarjan algorithm to find all *bridges* of the graph [77]. In a graph G if there is a path from a vertex u to a vertex v but every path from u to v contains edge e, then we say e is a *bridge* of G. Identifying bridges is important as we show that they play an essential role on the running times of information spreading algorithms. Let us define a *k-whisker* to be the maximal subgraph that can be detached from the rest of network

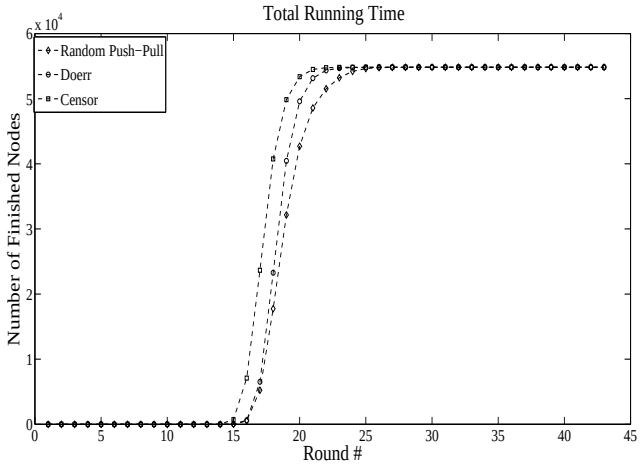


Figure 7.4: Running times of random push-pull, Doerr, and Censor algorithms without 1-whiskers

by removing k number of edges [56]. In particular, a *1-whisker* is a subgraph that is connected to the rest of network just by a *single edge*. A component is *k-edge connected* if we need to cut at least k edges to make the component disconnected. By removing all identified bridges, we can find the largest 2-edge connected component (i.e. the *core* of G). Having the core of the graph and all bridges, we can identify the 1-whiskers by only removing those bridges that have one node in the core. Identifying 1-whiskers is crucial because they are small components that are weakly connected to the core. Figure 7.2 shows a subset of identified 1-whiskers in the Facebook friendship graph. As we can see, 1-whiskers have a variety of structures although they are connected to the rest of the network by a single edge.

We show the number of 1-whiskers as a function of their sizes in Figure 7.3. We find that around 7% of 1-whiskers have more than 2 nodes. In particular, we identify 7736 number of 1-whiskers where 590 of them have at least two nodes. Among these 590 1-whiskers we find a variety of structures. While some of them are paths, there are some 1-whiskers that have a rich internal structure as it is shown in Figure 7.2. Next, we remove all 1-whiskers from the Facebook graph to measure the importance of 1-whiskers on the performance of the spreading algorithms. The running times of the random push-pull, Doerr, and Censor algorithms without 1-whiskers are shown in Figure 7.4. Comparing Figures 7.1 and 7.4, we can see that removing 1-whiskers decreases the running times of the random push-pull and Doerr algorithms by at least a factor of two. Our results show that there are a significant number of bridges

in social network that act as the main communication bottlenecks for information spreading.

7.3.4 An Efficient Framework for Finding All k-cuts

A cut of graph $G = (V, E)$ is a partition of its veritces (i.e. V) into two disjoint subsets. The *cut-set* of a cut is the set of edges whose endpoints are in different subsets of the partition. A *k-cut* is a cut whose cut-set has size k. After we remove all 1-whiskers, we have a graph that is 2-edge connected. The next step is to find all maximal 2-edge connected components that can be detached from the rest of network by removing two edges (i.e. 2-whiskers). This process can be continued until we find a core that is very well-connected. Hence, we need an algorithm to find all k-cuts of a graph. Given a weighted directed graph G, a *max-flow* algorithm finds a way of transporting the maximum amount of the given commodity from a vertex s to a vertex t [30]. By running a max-flow algorithm between two given nodes s and t, we can compute the size of *s-t cut* (i.e. the minimum number of edges that we need to cut to disconnect s from t). Thus, we can figure out the connectivity of a graph in details by running the max-flow between all possible $\binom{n}{2}$ pairs of vertices. This is computationally intensive as our social network is very large. However, Gusfield showed that we just need to run n instances of max-flow algorithm to find all pairs max-flow of G [33]. He proposed a simple and efficient method that constructs a *flow tree* by running max-flow algorithm for n times. The resulting flow tree has all possible $\binom{n}{2}$ max-flow results.

It is possible to study the detailed connectivity properties of the social network by analyzing the computed flow tree. We implemented the Gusfield algorithm to compute the flow tree of the Facebook graph. The efficiency of Gusfield algorithm directly depends on the algorithm that we use for computing the max-flow between two given nodes. We used the Edmonds-Karp algorithm that is a specialization of Ford-Fulkerson [30]. We run Gusfield's algorithm on the largest 2-edge connected component computed from the previous part to find the connectivity of the social network.

As it has been mentioned, the largest component of the original social network has $|V| = 63392$ nodes and $|E| = 817090$ edges. After we remove the first periphery layer (i.e. the 1-whiskers), we find the largest 2-edge connected component which has $|V_{2-edge}| = 54825$ and $|E_{2-edge}| = 808260$. In other words, the first peripheral layer of

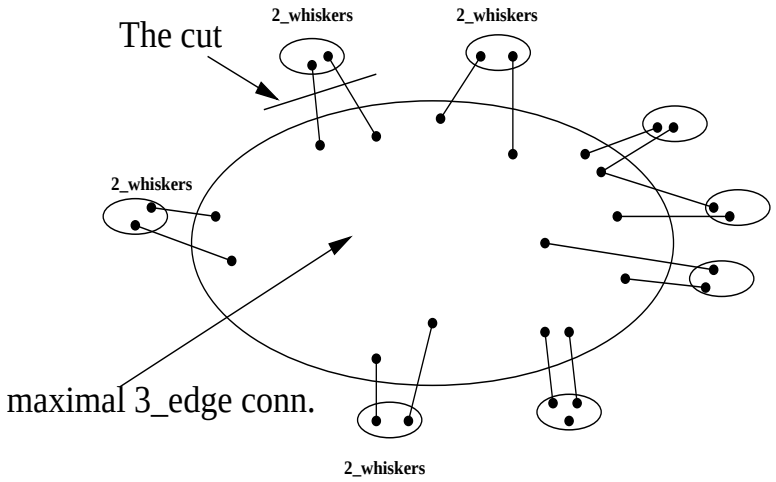


Figure 7.5: Computing the maximal 3-edge connected component

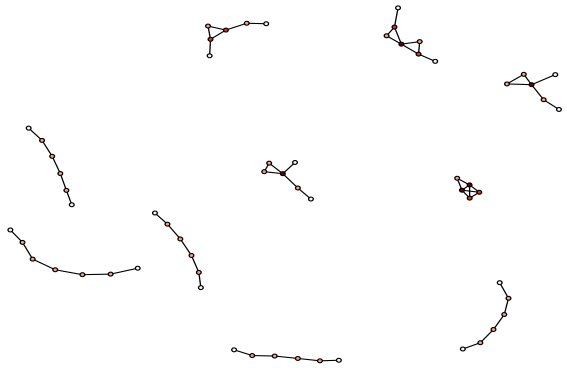


Figure 7.6: The first 10 largest 2-whiskers and their corresponding nodes of the core

the graph has 8567 nodes and 8830 edges.

To find the largest 3-edge connected component we have to find and cut all 2-whiskers from the remaining graph as shown in Figure 7.5. We compute the largest 3-edge connected component by removing all edges that have weight of two on the computed flow tree. If we remove all of these edges from the flow tree, the leftover is a tree where all nodes are connected by at least three edges that is the largest 3-edge connected component. This way we can compute the 3-edge connected core and all of 2-whiskers. We find that the next peripheral layer has 4991 nodes and 9669 edges and the largest 3-edge connected component has $|V_{3-edge}| = 49834$ and $|E_{3-edge}| = 798591$.

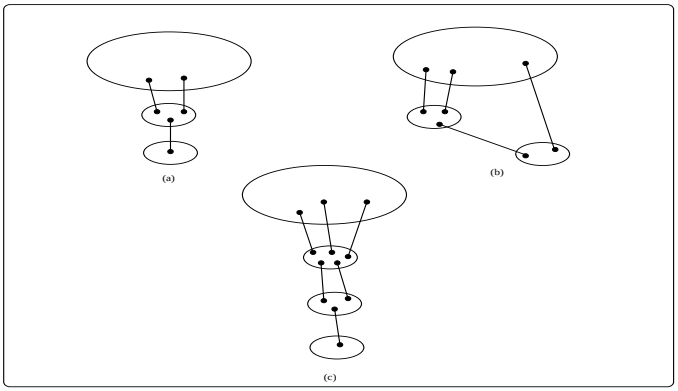


Figure 7.7: Components connectivity in a social network

The 2-whiskers are less sparse than 1-whiskers as they are 2-edge connected components. In other words, as we cut the periphery of the graph layer by layer, the core of the graph becomes denser and denser. We draw the 10 largest 2-whiskers in Figure 7.6. Although the internal connectivity of 2-whiskers components are far from cliques, some of them have richer structure than trees.

Figure 7.3 shows the number of 2-whiskers as a function of their sizes. A large number of 2-whiskers are single nodes that are connected to the rest of the graph by only two edges. The size and the internal structure of 2-whiskers directly influence the speed of information spreading in social networks. Since 2-whisker components are connected to the rest of the graph just by two edges, the information leaves and enters them slowly. Moreover, as the size of 2-whiskers grows, the running time of information spreading algorithm increases. Figure 7.3 shows that around 7% of 2-whiskers have more than 2 nodes. We have found 4647 number of 2-whiskers where 284 of them have at least two nodes.

The k-whiskers can be connected to each others through a chain as shown in Figure 7.7 (a) and (c). However, the connectivity among k-whiskers can be more complex than a simple chain as it is shown in Figure 7.7 (b). To figure out the connectivity among k-whiskers we need to run Gusfield's algorithm recursively. In Figure 7.7 (b) we can see that after we remove the 2-whisker by running Gusfield algorithm, it influences the connectivity of the 3-whisker as it turns the 3-whisker component into a 2-whisker. Therefore, every time that we run Gusfield algorithm, it changes the structure of the graph. Thus, it is necessary to run it again if we want to find the 3-whisker components.

7.3.5 Discussion

We studied the statistical properties of communities in an undirected graph collected from the Facebook website. We identified communities by using graph conductance definition. We found that the best communities have relatively small sizes that are in the order of 10 nodes (i.e. 1-whiskers). Most importantly, we found that Facebook graph has a core-periphery structure. While the core of the Facebook graph is well-connected, there are a large number of 1-whisker components that are connected just with a single edge to the core. Our empirical results also show that these 1-whiskers play a fundamental role in slowing down the speed of information spreading algorithms. Our results are in agreement with Leskovec et al. observations in several large social networks [56].

7.4 Mathematical Analysis

In this section, we study the effect of 1-whiskers on running times of the three spreading algorithms. We postulate a core-periphery model for community structure of social networks based on our empirical results from the previous section. Our main focus is to develop a mathematical model which explains our empirical results for running times of the information spreading algorithms.

7.4.1 The Effect of 1-whiskers on Information Spreading

We observed a variety of structures for 1-whiskers. In particular, we found that some of 1-whiskers have path structure. From our Facebook data we also saw that the size of 1-whiskers are bounded by $O(\log n)$ which is in agreement with Leskovec et al. observation [56]. Lemma 1 shows the expected number of rounds that are required to spread messages through those 1-whiskers that have path-like structure. The length of a path is defined as the number of nodes in the path.

Lemma 1. *To spread information in a path-like 1-whisker of length L: (I) the random push-pull algorithm needs $2(L-1)$ expected number of rounds, (II) the Censor and Doerr algorithms in the worst case require $2L$ number of rounds.*

Proof. For the standard random push-pull the expected number of required steps for a message of one of the endpoints of a path to reach the other endpoint of the path is $2(L-1)$ (i.e. tossing a coin for each edge with $p = 0.5$). Hence, the expected

number of rounds is $\Theta(L)$. For the Censor algorithm, suppose u and v are neighbors in a path of length L. There is not any way for u to hear about v from its other neighbor. Therefore, there are two possible scenarios: either (1) u receives $m(v)$ when it contacts v directly for the first time, or (2) u receives $m(v)$ when v contacts u for the first time. In the first case, the edge (u, v) turns black (i.e. u keeps v in its bottleneck list); however, in the second scenario (u, v) turns white (i.e. u removes v from its bottleneck list) but instead (v, u) turns black because v receives $m(u)$ for the first time from u itself when v directly contacts u. We can extend the same argument to all nodes in the path. Therefore, in the worst case after the first two steps, there is a continuous black path from one of the path end node to the other end node. This guarantees that at most we need $2L$ steps to spread all messages of a path to every node in the path. The Doerr algorithm gives us the same performance as the Censor because in the first step each node contacts one of its neighbors on the path randomly. However, after the first step nodes start contacting their neighbors deterministically by alternating between their two neighbors because each node on a path maximally has two neighbors. Therefore, in the worst case the Doerr algorithm also needs $2L$ rounds to finish. □

From Lemma 1, the number of required steps for the Censor and Doerr algorithms to spread messages in path-like 1-whiskers in the worst case is proportional to the length of the 1-whiskers (i.e. $\log n$). However, the expected number of steps for the random push-pull is proportional to the length of the path. This means that the number of required steps for the random push-pull to spread messages inside a path-like 1-whisker can be larger than the expected value. This becomes especially important when we consider the fact that the number of 1-whiskers is proportional to the size of the network (i.e. a constant fraction of n) [56].

Based on our empirical analysis, we can identify two different types of 1-whiskers as it is shown in Figure 7.8. Although a *Type I* 1-whisker is connected with an edge (u, v) to the core where $deg(u) = 2$ and $deg(v) = \Theta(n)$, a *Type II* 1-whisker is connected with an edge (u, v) to the core where $deg(u) = \Theta(\log n)$ and $deg(v) = \Theta(n)$. The structures of Type I 1-whiskers are similar to paths; however, Type II 1-whiskers are far from paths, and they are internally well connected. If we assume that 1-whiskers of Type II are similar to cliques, then the expected number of steps for spreading messages inside them for all three strategies is $\Theta(\log l)$ where l denotes the size of a Type II 1-whisker. This means that we need $\Theta(\log \log n)$ steps to spread

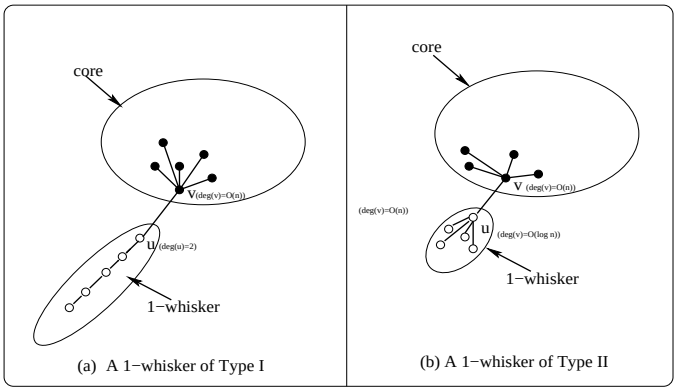


Figure 7.8: Two types of 1-whiskers identified from empirical data

messages of a Type II 1-whiskers internally which is quite low. Lemma 1 states that the running times of both Censor and Doerr algorithms for spreading messages inside path-like 1-whiskers are quite similar. However, our empirical results show that the Censor strategy performs better than Doerr algorithm in the presence of 1-whiskers.

To justify our empirical results for running times of the Censor and Doerr algorithms, we need to consider 1-whiskers of Type II. As it has been mentioned earlier, the number of steps for spreading messages inside a Type II 1-whiskers is similar for all algorithms (i.e. $\Theta(\log \log n)$). However, the main difference is the number of required steps for a node u inside of a 1-whisker to pass its component's messages to a node v inside the core or vice versa. Let us assume that $deg(v) = \Theta(n)$ where $deg(.)$ denotes the degree of a node. This assumption is reasonable because nodes inside the core are well connected to the other nodes in the core. Let $C_{1-whisk}$, $C^{T1}_{1-whisk}$, and $C^{T2}_{1-whisk}$ denote the set of all 1-whiskers, 1-whiskers of Type I, and 1-whiskers of Type II, respectively. Let also denote the total number of required rounds for Type I and Type II 1-whiskers to pass their messages to the core by T^1_{cross} and T^2_{cross}, respectively. In what follows, we first consider the T^1_{cross} for 1-whiskers of Type I and then we analyze the T^2_{cross} for Type II 1-whiskers.

Lemma 2. *For Type I 1-whiskers: (I) the Doerr algorithm in the worst case needs* $T^1_{cross} = \Theta(1)$ *number of rounds, (II) the Censor algorithm in the worst case needs* $T^1_{cross} = \Theta(1)$ *number of rounds with a high probability, and (III) the random push-pull needs* $E[T^1_{cross}] = \Theta(\log(|C^{T1}_{1-whisk}|))$ *expected number of rounds to send messages of Type I 1-whiskers to the core of the graph.*

Proof. For Type I, by using the same argument as Lemma 1 one can show that Doerr algorithm needs a constant number of steps to pass the messages of all 1-whisker

to the core of the network. In the case of the Censor algorithm, we are to show that it is very likely for node u to receive the message of its corresponding node v from the core for the first time when u contacts v. If this happens, edge (u, v) turns black and node u stores node v in its bottleneck list. As a result, it takes a constant number of rounds (i.e. $\Theta(1)$) for node u to exchange its messages with node v. If a Type I 1-whisker has size one, it is clear that edge (u, v) turns black. Let $C^{T1'}_{1-whisk}$ denote the set of all 1-whiskers of Type I with size greater than one. Let $p_b(u, v)$ also denote the probability that the directed edge (u, v) turns black. For every node v inside the core and its corresponding node u from 1-whisker, we can show that $p_b(v, u) = \frac{1}{n+1}$. Thus, we have $p_b(u, v) = \frac{n}{n+1}$. Let $p^{T1'}_{1-whisk}$ denote the probability that all edges (u, v), from Type I 1-whiskers with size greater than one to the core, turn black. We have $p^{T1'}_{1-whisk} = (1 + \frac{1}{n})^{-|C^{T1'}_{1-whisk}|}$. Our empirical results show that $|C^{T1'}_{1-whisk}| = 510 \ll n = 63k$. Hence, we have $p^{T1'}_{1-whisk} \approx e^{-\frac{|C^{T1'}_{1-whisk}|}{n}}$, which is very close to one for large n's. As a result, the Censor algorithm also requires $\Theta(1)$ number of rounds to send over messages of all 1-whiskers to the core of the graph with a high probability.

For the random push-pull, let us denote by X_{uv} the number of required rounds for one of the nodes u or v to pass its message to the other one. X_{uv} is a geometric random variable with probability of $1 - \frac{1}{2}(1 - \frac{1}{n})$. Since all of 1-whiskers have to send their messages to the core successfully, we have $E[T^1_{cross}] = E[\max X_{uv}]$ for all pairs of nodes such as u and v that are the endpoints of all bridges that connect 1-whiskers to the core of G. Given n independent geometric random variables X_i's, each with success probability of p, it is known that $E_n = E[\max_{1 \le i \le n} X_i] \approx \frac{1}{2} + \frac{-1}{\log(1-p)} H_n$, where H_n is the n^{th} harmonic number $H_n = \sum_{k=1}^{n} \frac{1}{k}$ [24]. Therefore, for the random push-pull algorithm we have $T^1_{cross} = \Theta(\log(|C^{T1}_{1-whisk}|))$. □

So far, we have not found any difference between number of required steps for Censor and Doerr algorithms considering the Type I of 1-whiskers. In Figure 7.8 (b), while node u inside a 1-whisker has a degree of $\Theta(\log n)$, node v inside the core has a linear degree. This assumption is reasonable considering the fact that the size of 1-whiskers is $O(\log n)$.

Lemma 3. *For Type II 1-whiskers: (I) the random push-pull and Doerr algorithms need* $E[T^2_{cross}] = \Theta(\log(|C^{T2}_{1-whisk}|) \log n)$ *expected number of rounds and (II) the Censor algorithm in the worst case needs* $T^2_{cross} = \Theta(\log \log n)$ *number of rounds with a high probability to send messages of Type II 1-whiskers to the core of the graph.*

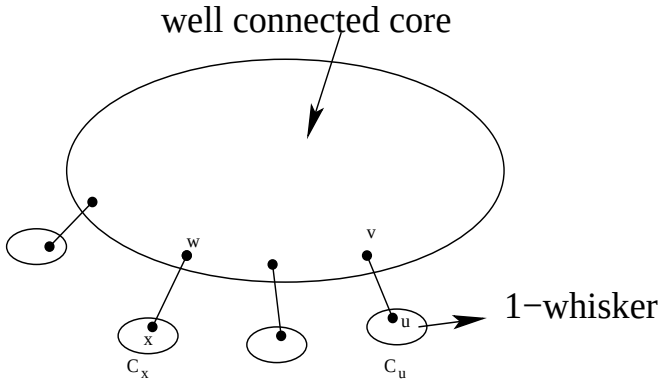


Figure 7.9: Well-connected core and weakly connected periphery

Proof. For Type II 1-whiskers, node u in a 1-whisker has degree of $\Theta(\log n)$ while node v inside the core has degree of $\Theta(n)$. Similar to Lemma 2, we are to show that it is very likely for node u to receive the message of its corresponding node v from the core for the first time when u contacts v. If this happens, edge (u, v) turns black and node u stores node v in its bottleneck list. As a result, it takes $2|Bot(u)| = \Theta(\log \log n)$ number of rounds for node u to pass its messages to node v. This guarantees that for the Censor algorithm we have $T^2_{cross} = \Theta(\log \log n)$. We can show that $p_b(v, u) = \frac{\log n}{n+\log n}$. Thus, we have $p_b(u, v) = \frac{n}{n+\log n}$. Let $p^{T2}_{1-whisk}$ denote the probability that all edges (u, v) from Type II 1-whiskers to the core turn black. We have $p^{T2}_{1-whisk} = (1 + \frac{\log n}{n})^{-|C^{T2}_{1-whisk}|}$. Our empirical results show that $|C^{T2}_{1-whisk}| = 80 \ll n = 63k$. Hence, for sufficiently large n we have $p^{T2}_{1-whisk} \approx e^{-\frac{|C^{T2}_{1-whisk}| \log n}{n}}$, which is very close to one. As a result, the Censor algorithm requires $\Theta(\log \log n)$ number of rounds to send messages of Type II 1-whiskers to the core of the graph with a high probability.

However, the Doerr algorithm cannot do much better than the random push-pull where the probability that in a given round either u or v contact the other one is $1 - (1 - \frac{1}{\log n})(1 - \frac{1}{n})$. Considering the fact that all 1-whiskers should successfully send over their messages, we have $E[T^2_{cross}] = E[\max X_{uv}]$ for all pairs of nodes such as u and v that are the endpoints of all bridges in the graph. Therefore, we have $T^2_{cross} = \Theta(\log(|C^{T2}_{1-whisk}|) \log n)$ for both random push-pull and Doerr algorithms. □

7.4.2 Hypothesis: A Core-Periphery Structure

To compute the overall running times of spreading algorithms, we need to postulate a model for our social network. Based on our empirical results as well as results of [56], we assume a core-periphery structure for our social network in which a large number of 1-whiskers are weakly connected to a very well-connected core. We already have found that 1-whiskers have a variety of structures. In the worst scenario, these components are paths with $O(\log n)$ nodes.

Theorem 1. *The total running time of the Censor algorithm is* $\Theta(\log n)$*; however, the total running time of the random push-pull and Doerr algorithms is* $\Theta(\log(|C^{T2}_{1-whisk}| \log n))$.

Proof. For the graph shown in Figure 7.9 the bottlenecks are 1-whisker components that are connected with a single edge to the core. Let T_{core} and T_{peri} denote the required number of rounds for spreading messages inside the core and 1-whiskers respectively. Let also show the total number of required rounds for all 1-whiskers to pass their messages to the core with T_{cross}. If we assume that the core of the graph is well-connected, we have $T_{core} = \Theta(\log n)$ for all three spreading algorithms. If we use the Censor algorithm, we have $T_{peri} = \Theta(\log n)$ as 1-whiskers with path-like structure take the longest time to spread their messages internally. We have also shown that in the worst case $T_{cross} = \Theta(\log \log n)$ by using the Censor strategy. To find the running time of the Censor algorithm, we need to compute the number of required rounds for spreading C_u's messages to a node in C_x where C_u and C_x are two distinct 1-whiskers as shown in Figure 7.9. Let T denote the total running time for the Censor algorithm. We find the running time as follows:

$$T = 2T_{peri} + 2T_{cross} + T_{core} = \Theta(\log n) \tag{7.3}$$

We repeat the same analysis for the Doerr algorithm for which we have already shown that $T_{peri} = \Theta(\log n)$ similar to the Censor algorithm. However, in the worst case $T_{cross} = \Theta(\log(|C^{T2}_{1-whisk}|) \log n)$. Therefore, we compute the total running time of the Doerr algorithm as below:

$$\begin{aligned} T &= 2T_{peri} + 2T_{cross} + T_{core} \\ &= \Theta(\log(|C^{T2}_{1-whisk}| \log n)) \end{aligned} \tag{7.4}$$

In the case of the random push-pull, we have already found that $T_{peri} = \Theta(\log n)$ while $T_{cross} = \Theta(\log(|C^{T2}_{1-whisk}| \log n))$. Thus, the total running time of random push-pull is also the same as the Doerr algorithm. □

Comparing equation 7.3 with 7.4 we see that the Censor algorithm performs a $\log(|C^{T2}_{1-whisk}|)$ factor better than the random push-pull and Doerr algorithms. Finally, Equations 7.3 and 7.4 are in agreement with our empirical results from the previous section.

7.4.3 Discussion

Our mathematical analysis showed that the Censor algorithm performs faster than the Doerr algorithm. This is mainly because the Censor algorithm works well in the presence of communication bottlenecks while the Doerr strategy does not. In particular, the Censor algorithm efficiently identifies the communication bottlenecks and chooses them with a higher probability than the other two algorithms. This strategy helps the Censor algorithm to perform better than the other two algorithms in the presence of 1-whiskers.

7.5 Graph Sparsification

It was shown that *inter-cluster* edges (i.e. edges that fall between clusters) play an important role in the speed of information spreading algorithms in social networks [70]. Thus, we label inter-cluster edges as important edges for information spreading purpose. *Graph sparsification* is a well-known technique for sparsifying dense graphs by throwing out unimportant edges. Graph sparsification was shown to be an effective technique for speeding up the running times of graph algorithms [73]. In this section, we sparsify our social graphs by throwing out intra-cluster edges (i.e. edges that fall inside clusters) as unimportant edges. We store the important edges in bottleneck lists and visit them in odd rounds sequentially similar to the Censor algorithm while we follow the same strategy as the random push-pull in even rounds for information spreading.

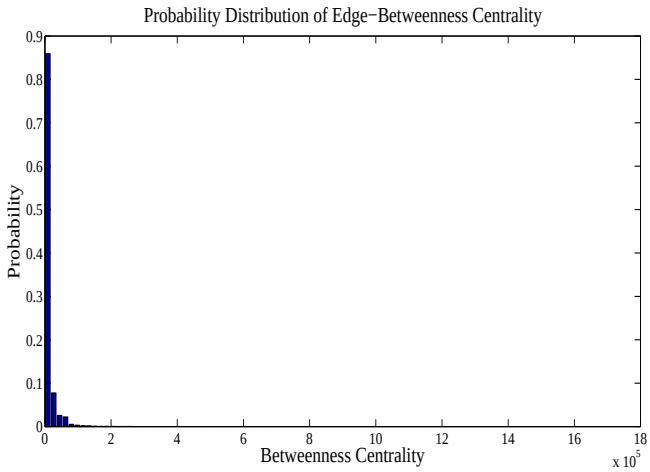


Figure 7.10: The edge-betweenness centrality distribution in Facebook

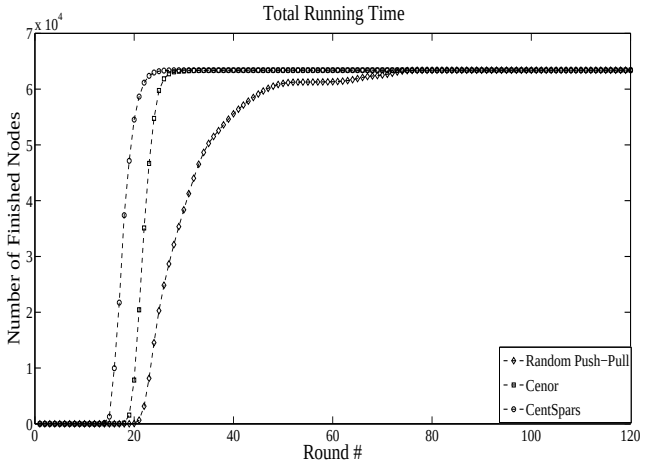


Figure 7.11: Information spreading in Facebook by using the centrality sparsification technique

7.5.1 Sparsification using Betweenness Centrality

To identify inter-cluster edges, we use the edge-betweenness centrality measure. In this section, we compute the exact betweenness centrality for all edges in order to sparsify our social graph for fast information spreading. First, let us define the betweenness centrality for a given edge $e \in E$ from graph $G = (V, E)$ as follows [26]:

$$Bet(e) = \sum_{(u,v) \in V \times V} \delta(u, v; e), \tag{7.5}$$

where $Bet(e)$ represents the betweenness centrality of edge e and $\delta(u, v; e)$ denotes the number of shortest paths from node u to v in the graph G which contain edge e.

It was shown that edges with high betweenness centralities act as bridges and fall between clusters [68]. This is our main motivation to compute the centrality of edges in order to identify bottleneck edges. It is important to mention that the identified bridges which connect 1-whisker components to the core of the Facebook graph have a large centrality in the order of $O(n \log n)$ where we assume that 1-whiskers sizes is $O(\log n)$. This confirms that betweenness centrality can correctly classify bridges as the bottlenecks. Figure 7.10 shows the probability distribution of edge-betweenness centrality in the Facebook graph. From the centrality distribution, we observe that most edges have low centralities. In particular, we find that over 80% of edges have a centrality less than $16k$ while the highest centrality is $1770k$ which belongs to an edge which interestingly falls in the core of the Facebook graph.

Now, we use the betweenness centrality of edges in order to select bottleneck edges (i.e. edges with high betweenness centralities) and store them in the bottleneck lists. To sparsify the graph, each node u sorts all of its neighbors according to their corresponding edge-betweenness centralities. Next, u stores one of its neighbors with the highest betweenness centrality in its bottleneck list. Our information spreading results are shown in Figure 7.11. We see that the centrality sparsification outperforms the random push-pull and Censor algorithms.

7.5.2 Discussion

Using the edge-betweenness centrality is helpful for information spreading as it allows us to identify important edges for information spreading. Furthermore, our results show that the sparsification of Censor algorithm works worse than the centrality spar-

sification because the Censor algorithm just uses local properties of graphs to identify bottlenecks whereas the betweenness centrality is a global optimization measure in the graph.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In the first part of this thesis, we employed the Jacard index to compute the similarity between social profiles of people. We next proposed a cost-effective routing algorithm inspired by Milgram's experiment and Kleinberg's small-world network model. By using the human contact traces collected from the Infocom 2006's conference, we compared the performance of our proposed Social-Greedy algorithm with other existing routing algorithms in the field. Our results show the applicability of bootstrapping wireless devices with people's social profiles for routing purposes in a conference setting. To the best of our knowledge, our work is the first empirical evaluation of Milgram's greedy strategy for routing messages to their destinations in a mobile social setting, and the first empirical evaluation for a routing strategy which employs only a social distance rather than a geographic distance to determine each forwarding decision.

Next, we exploited the small-world network properties of contact graphs and proposed three different predictors for inferring hidden contacts in partial contact graphs. Our prediction results show the efficiency of using social profiles and the small-diameter property of contact graphs for the prediction task. Motivated by the large number of external devices in previously collected contact traces, we also proposed several unsupervised predictors based on the neighborhood similarity in contact graphs and social similarity between people in order to reconstruct existing partial contact traces. We showed the importance of predicting missing contacts in completing cliques distribution in contact graphs. We also showed that combining the Foci

social similarity with the neighborhood similarity provides the best prediction results in terms of precision. Finally, we mathematically showed why the number of common neighbors outperforms other contact predictors.

Furthermore, we employed the machine learning approach for the contact prediction problem. We extracted several features from contact graphs structure and social profiles of people. We employed two supervised learning algorithms for the prediction task. Our results show that the number of common neighbors and the total overlap time are the most significant features for contact prediction. We also showed that nodes with high centrality are more predictable than nodes with low centrality. To the best of our knowledge, our work is the first attempt for formulating the contact prediction problem in the area of social networking.

In this thesis, we focused on predicting hidden and unobserved contacts for a few number of social settings such as two conferences, two campus environments, and one outdoor event. However, we strongly believe that both NCN and Foci-NCN predictors can also be used for predicting missing contacts for larger scale datasets because both methods rely on the time-spatial locality and the social similarity principles which play essential roles in the structure of human contact graphs.

Finally, we studied the importance of social networks structure on the speed of information spreading algorithms. For the first time, we showed that the Censor algorithm is the fastest distributed algorithm for information spreading in social networks. We justified this by identifying a large number of bridges in the Facebook graph which the Censor algorithm successfully detects them to some degree. We then showed that inter-cluster edges in social networks play an essential role in information spreading. In particular, we empirically showed that by identifying inter-cluster edges one can sparsify unimportant edges and speed up the rate of information spreading in social networks.

8.2 Future Work

One interesting question which can be studied as future work by using human mobility traces is to compare the effects of geographical positions of nodes with their social profiles on their mobility patterns. This implies that our proposed Social-Greedy routing may need to switch between geographic and non-geographic social information for routing based on the context of environment.

An appealing mobile social application would be the ability to detect an event

(e.g. a coffee break) by exploiting the distributions of contact durations and number of contacts. In particular, studying the correlation between an event and statistical properties of contact graphs and the possibility of taking advantage of this to extract the context of an ongoing event is another interesting direction to pursue as future work. Having access to the program schedules of Infocom 2005 and 2006 conferences and participants' contact traces, one should be able to test the correlation between properties of contact graphs and ongoing events in order to propose algorithms for predicting contexts of ongoing events.

Another interesting direction would be to propose a weighted version of our current social similarities by assigning different weights to distinct social dimensions according to the context. One can study the performance of such a weighted social similarity for predicting human contacts. Furthermore, proposing more efficient methods for predicting missing contacts in large geographical spaces as in MIT dataset is another promising area to be investigated. In Chapter 5, for our mathematical analysis, we assumed that nodes are uniformly distributed on the plane; however, more realistically one should expect nodes' locations to be influenced by social characteristics of people. Thus, we can introduce a *Geometric Social Network* model (GSN) instead of RGG. Characterizing nodes' distribution in GSN model is another open problem that can be studied in the future.

Bibliography

[1] Arab Spring. http://en.wikipedia.org/wiki/Arab_Spring. Accessed: 16/06/2012.

[2] Social-Sim Simulator. http://www.csc.uvic.ca/~jahan/codes/social-sim.html. Accessed: 13/08/2012.

[3] US Airways Flight 1549. http://en.wikipedia.org/wiki/US_Airways_Flight_1549. Accessed: 16/06/2012.

[4] Weka 3 - data mining with open source machine learning software, 2012.

[5] Lada A. Adamic and Eytan Adar. How to search a social network, November 2004.

[6] Lada A. Adamic, Rajan M. Lukose, Amit R. Puniyani, and Bernardo A. Huberman. Search in power-law networks. *Physical Review E*, 64(4):046135+, Sep 2001.

[7] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 519–528, New York, NY, USA, 2012. ACM.

[8] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[9] Chiristopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[10] Chiara Boldrini, Marco Conti, and Andrea Passarella. Exploiting users social relations to forward data in opportunistic networks: the hibop solution. *Pervasive and Mobile Computing (PMC)*, 2008.

[11] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[12] Keren Censor-Hillel and Hadas Shachnai. Fast information spreading in graphs with large weak conductance. In *SODA*, pages 440–448. SIAM, 2011.

[13] Meeyoung Cha, Alan Mislove, and Krishna P. Gummadi. A measurement-driven analysis of information propagation in the flickr social network. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 721–730. ACM, 2009.

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

[15] Xu Cheng, C. Dale, and Jiangchuan Liu. Statistics and social network of youtube videos. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*, pages 229–238, 2008.

[16] Flavio Chierichetti, Silvio Lattanzi, and Alessandro Panconesi. Rumor spreading in social networks. In *Proceedings of the 36th Internatilonal Collogquium on Automata, Languages and Programming: Part II*, ICALP '09, pages 375–386, Berlin, Heidelberg, 2009. Springer-Verlag.

[17] Flavio Chierichetti, Silvio Lattanzi, and Alessandro Panconesi. Almost tight bounds for rumour spreading with conductance. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 399–408, New York, NY, USA, 2010. ACM.

[18] Peter Csermely. *Weak Links: Stabilizers of Complex Systems from Proteins to Social Networks*. Springer Berlin Heidelberg, New York, 2006.

[19] Elizabeth M. Daly and Mads Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *MobiHoc '07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 32–40, New York, NY, USA, 2007. ACM.

[20] Peter S. Dodds, Roby Muhamad, and Duncan J. Watts. An experimental study of search in global social networks. *Science*, 301(5634):827–829, August 2003.

[21] Benjamin Doerr, Mahmoud Fouz, and Tobias Friedrich. Social networks spread rumors in sublogarithmic time. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, STOC '11, pages 21–30, New York, NY, USA, 2011. ACM.

[22] Nathan Eagle, Alex Pentland, and David Lazer. Inferring social network structure using mobile phone data. *Proceedings of the National Academy of Sciences (PNAS)*, 106(36):15274–15278, July 2009.

[23] David Easley and Jon Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

[24] B. Eisenberg. On the expectation of the maximum of IID geometric random variables. *Statistics & Probability Letters*, 78(2):135–143, February 2008.

[25] Michael Fire, Lena Tenenboim, Ofrit Lesser, Rami Puzis, Lior Rokach, and Yuval Elovici. Link prediction in social networks using computationally efficient topological features. In *SocialCom/PASSAT*, pages 73–80, 2011.

[26] L.C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[27] Debra S. Goldberg and Frederick P. Roth. Assessing experimentally derived interactions in a small world. *Proceedings of the National Academy of Sciences (PNAS)*, 100(8):4372–4376, February 2003.

[28] Jacob Goldenberg, Barak Libai, and Eitan Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 12:211–223, 2001.

[29] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 1019–1028, New York, NY, USA, 2010. ACM.

[30] Michael T. Goodrich and Roberto Tamassia. *Algorithm Design*. John Wiley Inc., 2002.

[31] M.S. Granovetter. The Strength of Weak Ties. *The American Journal of Sociology*, 78(6):1360–1380, 1973.

[32] Piyush Gupta and P. R. Kumar. Critical power for asymptotic connectivity in wireless networks. *In Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming*, pages 547–566, 1998.

[33] Dan Gusfield. Very simple methods for all pairs network flow analysis. *SIAM J. Comput.*, 19:143–155, February 1990.

[34] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

[35] Klaus Herrmann. Modeling the sociological aspects of mobility in ad hoc networks. In *MSWIM '03: Proceedings of the 6th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, pages 128–129, New York, NY, USA, 2003. ACM.

[36] Theus Hossmann, Thrasyvoulos Spyropoulos, and Franck Legendre. Know thy neighbor: Towards optimal mapping of contacts to social graphs for DTN routing. In *Proceedings of IEEE INFOCOM 2010*, pages 1–9. IEEE, March 2010.

[37] Wei-Jen Hsu, Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Ahmed Helmy. Modeling Spatial and Temporal Dependencies of User Mobility in Wireless Mobile Networks. *IEEE/ACM Transactions on Networking*, 17(5):1564–1577, October 2009.

[38] Pan Hui and Jon Crowcroft. Bubble rap: Forwarding in small world dtns in ever decreasing circles. Technical report, University of Cambridge, May 2007.

[39] Pan Hui and Jon Crowcroft. How small labels create big improvements. In *PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 65–70, Washington, DC, USA, 2007. IEEE Computer Society.

[40] Pan Hui and Jon Crowcroft. Human mobility models and opportunistic communication system design. *Royal Society Philosophical Transactions*, 366(1872), June 2008.

[41] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *MobiHoc '08: Proceedings of the 9th ACM inter-*

national symposium on Mobile ad hoc networking and computing, pages 241–250, New York, NY, USA, 2008. ACM.

[42] Pan Hui, Eiko Yoneki, Shu Y. Chan, and Jon Crowcroft. Distributed community detection in delay tolerant networks. In *MobiArch '07: Proceedings of first ACM/IEEE international workshop on Mobility in the evolving internet architecture*, pages 1–8, New York, NY, USA, 2007. ACM.

[43] Paul Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.

[44] Kazem Jahanbakhsh, Valerie King, and Gholamali C. Shoja. Empirical comparison of information spreading algorithms in the presence of 1-whiskers. In *SocialCom/PASSAT*, pages 489–492, 2011.

[45] Kazem Jahanbakhsh, Valerie King, and Gholamali C. Shoja. Predicting human contacts in mobile social networks using supervised learning. In *Proceedings of the Fourth Annual Workshop on Simplifying Complex Networks for Practitioners*, SIMPLEX '12, pages 37–42, New York, NY, USA, 2012. ACM.

[46] Kazem Jahanbakhsh, Valerie King, and Gholamali C. Shoja. Predicting missing contacts in mobile social networks. *Pervasive and Mobile Computing*, 2012.

[47] Kazem Jahanbakhsh, Gholamali C. Shoja, and Valerie King. Human contact prediction using contact graph inference. In *International Symposium on Social Computing and Networking*, pages 813–818. IEEE, December 2010.

[48] Kazem Jahanbakhsh, Gholamali C. Shoja, and Valerie King. Social-greedy: a socially-based greedy routing algorithm for delay tolerant networks. In *MobiOpp '10: Proceedings of the Second International Workshop on Mobile Opportunistic Networking*, pages 159–162, New York, NY, USA, 2010. ACM.

[49] Kazem Jahanbakhsh, Gholamali C. Shoja, and Valerie King. Predicting missing contacts in mobile social networks. In *World of Wireless Mobile and Multimedia Networks*, pages 1–9. IEEE, June 2011.

[50] K. Juszczyszyn, K. Musial, and M. Budka. Link prediction based on subgraph evolution in dynamic social networks. In *Privacy, security, risk and trust (pas-*

sat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom), pages 27 –34, oct. 2011.

[51] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, New York, NY, USA, 2003. ACM.

[52] Jon Kleinberg. The small-world phenomenon: an algorithm perspective. In *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 163–170, New York, NY, USA, 2000. ACM.

[53] Jon Kleinberg. Small-world phenomena and the dynamics of information. In *In Advances in Neural Information Processing Systems (NIPS) 14*, page 2001. MIT Press, 2001.

[54] Jérémie Leguay, Anders Lindgren, James Scott, Timur Friedman, and Jon Crowcroft. Opportunistic content distribution in an urban setting. In *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, CHANTS '06, pages 205–212, New York, NY, USA, 2006. ACM.

[55] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 641–650, New York, NY, USA, 2010. ACM.

[56] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Statistical properties of community structure in large social and information networks. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, pages 695–704, New York, NY, USA, 2008. ACM.

[57] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, New York, NY, USA, 2003. ACM.

[58] David Liben-Nowell and Jon Kleinberg. Tracing information flow on a global scale using Internet chain-letter data. *Proceedings of the National Academy of Sciences*, 105(12):4633–4638, March 2008.

[59] David Liben-Nowell, Jasmine Novak, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Geographic routing in social networks. *Proceedings of the National Academy of Sciences of the United States of America*, 102(33):11623–11628, August 2005.

[60] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.

[61] Miller McPherson, Lynn S. Lovin, and James M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

[62] Andrew Miklas, Kiran Gollu, Kelvin Chan, Stefan Saroiu, Krishna Gummadi, and Eyal de Lara. Exploiting social interactions in mobile systems. In *Proceedings of UbiComp 2007*, pages 409–428, September 2007.

[63] Stanely Milgram. The small world problem. *Psychology Today*, 1:60–67, 1967.

[64] Damon Mosk-Aoyama and Devavrat Shah. Computing separable functions via gossip. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, PODC '06, pages 113–122, New York, NY, USA, 2006. ACM.

[65] Abderrahmen Mtibaa, Augustin Chaintreau, Jason LeBrun, Earl Oliver, Anna-Kaisa Pietilainen, and Christophe Diot. Are you moved by your social network application? In *Proceedings of the first workshop on Online social networks*, WOSN '08, pages 67–72, New York, NY, USA, 2008. ACM.

[66] Mirco Musolesi and Cecilia Mascolo. A community based mobility model for ad hoc network research. In *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks*, REALMAN '06, pages 31–38, New York, NY, USA, 2006. ACM.

[67] Alex (Sandy) Pentland Nathan Eaglea and David Lazer. Inferring friendship network structure by using mobile phone data. *Proceedings of the National Academy of Sciences*, 106(36):15274–15278, September 2009.

[68] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, February 2004.

[69] Erik Nordström, Christophe Diot, Richard Gass, and Per Gunningberg. Experiences from measuring human mobility using bluetooth inquiring devices. In *MobiEval '07: Proceedings of the 1st international workshop on System evaluation for mobile platforms*, pages 15–20, New York, NY, USA, 2007. ACM.

[70] J. P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, and A. L. Barabási. Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences*, 104(18):7332–7336, May 2007.

[71] Mathew Penrose. *Random Geometric Graphs*. Oxford University Press, 2003.

[72] Everett M. Rogers. *Diffusion of Innovations*. Free Press, New York, 2003.

[73] Venu Satuluri, Srinivasan Parthasarathy, and Yiye Ruan. Local graph sparsification for scalable clustering. In *Proceedings of the 2011 international conference on Management of data*, SIGMOD '11, pages 721–732, New York, NY, USA, 2011. ACM.

[74] James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot, and Augustin Chaintreau. CRAWDAD trace for infocom2006. http://crawdad.cs.dartmouth.edu, May 2009.

[75] V. Sivaraman, S. Grover, A. Kurusingal, A. Dhamdhere, and A. Burdett. Experimental study of mobility in the soccer field with application to real-time athlete monitoring. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2010 IEEE 6th International Conference on*, pages 337 –345, October 2010.

[76] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-László Barabási. Limits of Predictability in Human Mobility. *Science*, 327(5968):1018–1021, February 2010.

[77] Robert Endre Tarjan. A note on finding the bridges of a graph. *Inf. Process. Lett.*, pages 160–161, 1974.

[78] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. Dias de Amorim, and J. Whitbeck. The Accordion Phenomenon: Analysis, Characterization, and Impact on DTN Routing. In *IEEE INFOCOM 2009 - The 28th Conference on Computer Communications*, pages 1116–1124. IEEE, april 2009.

[79] Amin Vahdat and David Becker. Epidemic routing for partially-connected ad hoc networks. Technical report, Duke University, April 2000.

[80] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, WOSN '09, pages 37–42, New York, NY, USA, 2009. ACM.

[81] Long Vu, Quang Do, and Klara Nahrstedt. Jyotish: Constructive approach for context predictions of people movement from joint wifi/bluetooth trace. *Pervasive and Mobile Computing*, 7(6):690 – 704, 2011.

[82] Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi. Human mobility, social ties, and link prediction. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 1100–1108, New York, NY, USA, 2011. ACM.

[83] D. J. Watts, P. S. Dodds, and M. E. J. Newman. Identity and search in social networks. *Science*, 296(5571):1302–1305, 2002.

[84] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

[85] Chang Wu Yu. Computing subgraph probability of random geometric graphs with applications in quantitative analysis of ad hoc networks. *IEEE J.Sel. A. Commun.*, 27(7):1056–1065, 2009.