Peer-to-peer Communication in Mobile Social Network

by

Zunnun A. Raoof Narmawala

(200921004)

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree

of

Doctor of Philosophy

in

Information and Communication Technology

to

Dhirubhai Ambani Institute of Information and Communication Technology

JULY 2016

Declaration

This is to certify that

1. the thesis comprises my original work towards the degree of Doctor of Philosophy in Information and Communication Technology at DA-IICT and has not been submitted elsewhere for a degree,

2. due acknowledgment has been made in the text to all other material used.

Signature of Student

Certificate

This is to certify that the thesis work entitled "Peer-to-peer Communication in Mobile Social Network" has been carried out by Zunnun A. Raoof Narmawala (200921004) for the degree of Doctor of Philosophy in Information and Communication Technology at this Institute under my supervision.

Thesis Supervisor Prof. Sanjay Srivastava

Abstract

Peer-to-peer opportunistic communication between mobile devices carried by humans without using any infrastructure is largely unexploited. As these devices are carried by humans, their encounter pattern depends on human mobility pattern which is governed by human social behaviour. Individuals belong to multiple communities, i.e. communities in social network overlap. These social ties significantly affect people's movement pattern and in-turn contact pattern of mobile nodes carried by them. This network paradigm is known as Mobile Social Network (MSN). The communication framework for MSN includes community-based and heterogeneous node popularity-based message transmission mechanisms. In this work, we design algorithms for such framework and develop two proof of concept applications on top of this framework. We also propose a realistic mobility model for MSN for reliable performance analysis through simulation.

Traditional mobility models, such as Random Way Point (RWP) and Brownian Motion (BM), model device mobility as random. However, researchers have shown that human mobility is rarely random and such models do not provide a reliable analysis of network protocol performance. Various characteristics of human mobility are derived in the literature from mobility traces and social network theory. We propose Community Aware Heterogeneous Human Mobility (CAHM) model incorporating all such characteristics. CAHM model is based on Heterogeneous Human Walk (HHW)[1] mobility model. CAHM achieves heterogeneous local popularity as observed in real mobility traces which HHW fails to achieve. It also incorporates following additional properties of human mobility: preference of nearby locations, speed as a function of distance to be traveled and power-law distributed pause time. We also analyze the performance of traditional and socially-aware routing protocols with CAHM and other existing mobility models. Our analysis shows that existing mobility models significantly underestimate the performance of socially-aware routing protocols.

Socially-aware routing protocols are designed to exploit the overlapping community structure of MSN for efficient communication. Nodes in MSN should be able to detect one or more communities in which it is member independently in a distributed manner because of the peer-to-peer nature of MSN. There is no distributed overlapping community detection mechanism for MSN proposed in the literature. We devise Distributed

Overlapping Community Detection (DOCD) mechanism by modifying non-overlapping community detection algorithm SIMPLE[2]. Simulation results show that DOCD detects overlapping community structure with 75-80% accuracy.

Analysis of properties of overlapping community structure formed by human mobility can give important insights for designing better routing protocols. We analyze, through simulation, properties such as actual community sizes, the probability of community existence and the fraction of hub and gateway nodes present in the community on an average. Different individuals have different local popularity within a community and different global popularity in the network. We call them hub and gateway nodes respectively. Our analysis of overlapping community structure establishes that small communities are transient. As per simulation results, the threshold for the same is around 10% of the total number of nodes in the network. Further, a higher fraction of hub and gateway nodes remain present in larger communities with less standard deviation as compared to smaller communities. Also, the fraction of gateway nodes present in a community is much lower than the fraction of hub nodes present in the community.

Identifying correct hub and gateway nodes is very important for the efficiency of a protocol aiming to exploit heterogeneous popularity. Existing methods to identify hub and gateway nodes require either flooding of messages or forwarding of not only node's encounter information but also accumulated encounter information from other nodes. We identify hub and gateway nodes using Markov-chain from the overlapping community structure itself (which is found using our DOCD mechanism) without doing message flooding or forwarding of accumulated encounter information from other nodes. Socially-aware routing protocols based on explicit community detection need to find such structure anyway for efficient forwarding. We make use of this structure for identifying hub and gateway nodes also. We compare ordered lists of hub and gateway nodes generated by Markov chain-based methods with ordered lists generated through simulation using flooding and encounter information. Simulation results show that these ordered lists are highly correlated, i.e. Markov chain-based methods correctly identify hub and gateway nodes.

We develop protocols for viral spread (many-to-all broadcast) and micro-blogging applications in MSN exploiting overlapping community structure and heterogeneous popularity of nodes for efficient forwarding. For the viral spread, buffering all packets from

all sources of a community at each node is prohibitive. Buffer space is limited and buffer occupancy level also has an impact on energy requirements. Existing protocols either work with specific assumptions regarding the type of MSN or do not aim to achieve improvement in terms of average packet delivery delay or delivery ratio in the presence of a limited buffer. In our Community Aware Viral Spread (CAVS) protocol, to reduce buffer usage, we probabilistically buffer received packets for forwarding. To offset the loss in performance due to this, we employ network coding. Network coding is a mechanism in which nodes encode two or more incoming packets and forward encoded packets instead of forwarding them as it is. Simulation results show that CAVS gives acceptable performance for buffering probability (P_b) value as low as 0.3 as compared to $P_b = 1$. Also, with a decrease in P_b, the performance of CAVS gets increasingly better than Epidemic routing[3]. For $P_b = 0.1$, packet delivery ratio and average packet delivery delay of CAVS is 122% more and 22% less than Epidemic routing respectively.

Our protocol for micro-blogging restricts message forwarding based on distance as well as time and keeps a constant amount of state information. The protocol is scalable and efficient as message filtering is done at the network protocol level itself as opposed to the conventional approach where it is done at the end points of the network. Further, because of in-network filtering, users can follow their interests and receive messages of their interests without getting overwhelmed, instead of having to identify and follow users with similar interests. It also eliminates the need to maintain a large amount of data at server machines. As server farms consume a huge amount of energy, it is useful to not to use infrastructure even though the bandwidth requirement of the application is limited. We also propose three models to generate user interest profiles synthetically where user interest profile is defined as a list of tags in which a user is interested along with interest level in each tag. Allen et al. in [4] have also proposed a micro-blogging protocol for MSN named as 'Uttering'. We measure and compare 'efficiency' and 'spread index' of our protocol with 'Uttering'. Efficiency is defined as the ratio of the total number of useful messages received by all nodes to the total number of messages transmitted by all nodes. Spread index is defined as the ratio of the total number of users who have received useful messages to the total number of users who were interested in those messages. Simulation results show that spread index of our protocol is better than 'Uttering' by 18-59% and efficiency is better than 'Uttering' by 35-56% for different user interest profile models.

To my parents, wife and kids for their love and support
which provided me the necessary strength to go through the rigours of Ph.D. work

Acknowledgements

I am deeply indebted to my thesis supervisor Prof. Sanjay Srivastava for his constant guidance, support, and motivation. He devoted a significant amount of his valuable time to plan and discuss my Ph.D. work. Without his insightful inputs, it would have been very difficult for me to do quality work. He is my role model as an academician. He has been a source of inspiration for me since I took his first course during my graduate study at DA-IICT.

I extend my gratitude to Prof. V. Sunitha and Prof. Anish Mathuria for their valuable suggestions during research progress seminars and during synopsis presentation respectively. I am also thankful to DA-IICT for providing me the opportunity to pursue Ph.D.

My special thanks to fellow Ph.D. student Manish Chaturvedi who was the first person I used to approach to discuss anything related to my Ph.D. work. His company kept me in good humour throughout.

Contents

List of Figures

List of Tables

List of Abbreviations

BM	Brownian Motion mobility model
CAHM	Community Aware Heterogeneous Human Mobility model
CAVS	Community Aware Viral Spread protocol
CCDF	Complementary Cumulative Distribution Function
CMM	Community-based Mobility Model
DOCD	Distributed Overlapping Community Detection
DTN	Delay Tolerant Network
HCMM	Home-cell Community-based Mobility Model
HHW	Heterogeneous Human Walk mobility model
JSI	Jaccard-based Similarity Index
LW	Levy Walk mobility model
MSN	Mobile Social Network
NS-2	Network Simulator-2
PDF	Probability Distribution Function
PSN	Pocket Switched Network
RLNC	Random Linear Network Coding
RWP	Random Way Point mobility model
SI	Susceptible Infected epidemic model
SLAW	Self-similar Least Action Walk mobility model
SNM	Social Network Model
SPoT	Social, sPatial and Temporal mobility framework
SWIM	Small World In Motion mobility model
TJSI	Threshold Jaccard-based Similarity Index

TTL	Time To Live
TVC	Time Variant Community mobility model
WDM	Working Day Mobility model
WWW	World Wide Web

List of Publications

Journal

1. Z. Narmawala and S. Srivastava, "Community aware heterogeneous human mobility (cahm): Model and analysis," in *Pervasive and Mobile Computing*. vol. 21, pp. 119–132, Elsevier 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574119214001977

   The publication is based on the work discussed in chapters 2, 4 and section 3.3. It is the extended version of the conference publication 1.

Conference

1. Z. Narmawala and S. Srivastava, "Improved heterogeneous human walk mobility model with hub and gateway identification," in *Distributed Computing and Networking*. pp. 469–483, Springer 2014. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-45249-9_31

   The publication is based on the work discussed in section 2.2 and chapter 4.

2. Z. Narmawala and S. Srivastava, "Viral spread in mobile social network using network coding," in *India Conference (INDICON), 2014 Annual IEEE*. IEEE, 2014, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=7030651

   The publication is based on the work discussed in chapter 5.

3. Z. Narmawala and S. Srivastava, "Scalable micro-blogging in mobile social communities," presented at *Region 10 Symposium, 2015 IEEE*. IEEE, 2015. [Online]. Accepted manuscript: https://sites.google.com/a/nirmauni.ac.in/zunnun/publications/paper.pdf?attredirects=0&d=1

   The publication is based on the work discussed in chapter 6.

Chapter 1

Introduction

In recent times, the growth of mobile devices (especially smartphones) is phenomenal. These devices support Bluetooth and Wifi connectivity. Further, they are equipped with good computing power and memory. Because of this, an entirely new network paradigm has emerged in which encounters between these mobile devices can be exploited for opportunistic data transfer without using any fixed network infrastructure[5]. This opportunistic network paradigm is called Pocket Switched Network (PSN) as we carry smartphones in our pockets. Message broadcast, news spread, traffic updates, micro-blogging and peer-to-peer file sharing are some of the applications which can run on such type of network. Further, it can significantly offload infrastructure based networks such as cellular network and infrastructure based Wifi network and reduce data cost for users considerably[6].

As these devices are carried by humans, their encounter patterns depend on human mobility patterns. Thus, knowledge of human movement behaviour and social structure can be exploited for efficient peer-to-peer communication[7, 8]. As a result, this network paradigm is also called Mobile Social Network (MSN).

1.1 Overlapping Community Structure in Mobile Social Network

Movements of individuals are not independent of each other. Humans belong to multiple social communities like friends, family, co-workers etc.[9, 10]. Individuals meet people

from the same community more frequently than people of other community[11] and some of these communities are tied to the corresponding places (locations). E.g. family and co-workers are tied to home and office respectively. So, these social ties significantly affect individuals' movement. The communities in a social network overlap as individuals belong to multiple communities[10], i.e. individuals also move between these communities. Further, some nodes meet more nodes in the community than others (*locally popular nodes*) or visit other communities more often than others (*globally popular nodes*). We call these locally and globally popular nodes as *hub* and *gateway* nodes respectively. These hub and gateway nodes can play important roles in spreading information in MSN efficiently.

The above community structure is called as *overlapping community structure* in MSN. As an example, consider a professor working in a university. Her office is in a faculty building. Professors sitting in that building form a community. She teaches a course to a group of students in a classroom. So, she is also part of those students' community. During office hours, she moves within and across these communities. Similarly, after office hours, she forms communities with family members and also with friends she meets in a park in the evening. It is evident from the example that the overlapping community structure changes as per different time periods of the day.

1.2 Mobility Modeling for Mobile Social Network

To analyze the performance of protocols which aim to exploit human movement behaviour through simulation, it is essential to design realistic mobility models which can mimic human mobility patterns as closely as possible. A number of experimental projects have been undertaken to collect encounter information of devices carried by humans[12, 13]. These traces can be used in the simulation to evaluate and analyze the performance of different protocols. While this approach generates realistic mobility patterns, its usefulness is limited as the performance of a protocol can be evaluated only for limited values of network parameters for which traces are available. Nonetheless, from analysis of these traces, various statistical properties of human mobility are derived[12, 13, 14, 15]. Well-known and widely used mobility models such as Random Way Point (RWP)[16], Brownian Motion (BM)[17] etc. do not exhibit these properties. So, trace-based mobility models such

as Levy Walk (LW)[14], TVC[18], SWIM[19], and SLAW[20] are proposed based on these statistical properties. Although these models are able to reproduce statistical properties of real-world mobility traces, they assume that each node moves independently of others.

Mobility models such as CMM[11] and HCMM[21] do model the community structure of humans but they do not model overlapping community structure and heterogeneous local and global popularity. These properties have a significant impact on the performance of forwarding strategy. Only Heterogeneous Human Walk (HHW) mobility model[1] attempts to incorporate these properties for generating overlapping community structure synthetically, i.e. it does not require real life social graph as an input, as requirement of real life social graph significantly restricts community structures that can be generated. Further, it also does not use Social Network Models (SNM) such as Caveman model[22] to generate community structure, as these models are quite simplistic and do not take into account all of the above social network theory based properties.

However, HHW does not incorporate following important trace-based properties: i) preference of nearby locations over far-away locations (Levy walk) ii) speed of nodes as a function of distance to be traveled and iii) power-law distributed pause time. Further, we show that HHW model does not correctly reproduce heterogeneous local popularity of nodes in a community as observed in mobility traces. So, we propose Community Aware Heterogeneous Human Mobility (CAHM) model which incorporates these properties and also generates correct heterogeneous local popularity.

1.2.1 Effect of Mobility Models on the Performance of Routing Protocols

We also analyze the performance of traditional and socially-aware routing protocols with CAHM and other existing mobility models. Our analysis shows that, compared to CAHM, existing mobility models significantly underestimate packet delivery ratio and average packet delivery delay of socially-aware routing protocols.

1.3 Distributed Overlapping Community Detection in MSN

To exploit overlapping community structure for efficient communication, nodes in MSN should be able to detect overlapping community structure in a distributed manner; i.e. each node should be able to detect one or more communities in which it is a member independently. SIMPLE, k-CLIQUE and MODULARITY algorithms proposed in [2] detect and maintain only one community per node. Further, in these algorithms, there is no mechanism to drop nodes from the community based on aging process. AD-SIMPLE [23] modifies SIMPLE and proposes aging mechanism to drop nodes from the community and maintains the current community only.

Another problem with MODULARITY, k-CLIQUE, SIMPLE, AD-SIMPLE and the mechanism in [24] is that they do not detect multiple communities of a node. But, nodes in MSN can belong to multiple communities like friends, family, co-workers etc. MODULARITY, k-CLIQUE, and SIMPLE merge all these communities and detect single community per node. AD-SIMPLE does not merge these communities but maintains only one of these communities as the current community of the node. It is desirable for a node to maintain information about all the communities in which it is a member. This information can be used by routing protocols to make better forwarding decisions. Further, as we shall see in chapter 4, this information is also used to detect hub and gateway nodes of communities without doing message flooding or forwarding of accumulated encounter information from other nodes.

Williams et al. in [25] propose a mechanism for decentralized detection of periodic encounter communities in opportunistic networks. Periodic encounter community means that a community is formed periodically when members of the community come in contact with each other. Then, it gets dissolved as its members move away from each other; e.g. during office hours, a community is formed every day between office workers which dissolves after office hours. So, this mechanism can detect different communities in which a node is a member in different periods. But, it is evident from the discussion in section 1.1, a node can be part of different communities within any given period and nodes can move between communities without communities getting dissolved. We call such a community structure as overlapping community structure. None of the existing mechanisms

detects overlapping community structure in MSN in a distributed manner. So, we propose Distributed Overlapping Community Detection (DOCD) mechanism by modifying SIMPLE algorithm to detect overlapping communities in a distributed manner. Simulation results show that DOCD detects overlapping community structure with 75-80% accuracy.

1.3.1 Analysis of Overlapping Community Structure

Analysis of properties of overlapping community structure formed by human mobility can give important insights for designing better communication protocols. We analyze, through simulation, properties such as actual community sizes, probability of community existence, and fraction of hub and gateway nodes present in the community on an average.

Our analysis of overlapping community structure establishes that small communities are transient. As per simulation results, the threshold for the same is around 10% of total number of nodes in the network. Further, a higher fraction of hub and gateway nodes remain present in larger communities with less standard deviation as compared to smaller communities. Also, the fraction of gateway nodes present in a community is much lower than the fraction of hub nodes present in the community. These results give important insights for designing better forwarding protocols for MSN.

1.4 Identifying Hub and Gateway Nodes in Overlapping Community Structure

Identifying correct hub and gateway nodes is very important for the efficiency of a protocol aiming to exploit heterogeneous popularity. Existing methods[5, 26, 27, 28] to identify hub and gateway nodes require either flooding of messages or forwarding of not only node's encounter information but also accumulated encounter information from other nodes. So, the performance of these methods does not scale with network size or community size. We identify hub and gateway nodes using Markov-chain from the overlapping community structure itself, which is found using our DOCD mechanism, without doing message flooding or forwarding of accumulated encounter information from other nodes. Socially-aware routing protocols based on explicit community detection need to find such structure

anyway for efficient forwarding[27, 29, 28, 5, 30, 31]. We make use of this structure for identifying hub and gateway nodes also. Further, in DOCD also, a node sends only its own encounter information to its neighbour and does not send accumulated encounter information from other nodes.

To validate Markov chain-based methods, we also propose two simulation-based approaches to identify hub nodes and another two simulation-based approaches to identify gateway nodes. We compare ordered lists of hub and gateway nodes generated by Markov chain-based methods with ordered lists generated by these methods. Simulation results show that these ordered lists are highly correlated, i.e. Markov chain-based methods correctly identify hub and gateway nodes.

1.5 Viral Spread in MSN

Viral spread (Many-to-all broadcast) in MSN is an important functionality which can be useful to a variety of applications in MSN such as message broadcast, news spread, traffic updates etc. Further, it is also useful for routing as many routing protocols require flooding of control messages. We propose a protocol called Community Aware Viral Spread (CAVS) for many-to-all broadcast in MSN. The protocol exploits properties of human mobility like overlapping community structure and heterogeneous popularity of nodes for efficient forwarding. Buffering all packets from all sources of a community at each node is prohibitive. Buffer space is limited and buffer occupancy level also has an impact on energy requirements[32]. Existing protocols[33, 34, 35, 36, 37] either work with specific assumptions regarding the type of MSN or do not aim to achieve improvement in terms of average packet delivery delay or delivery ratio in the presence of a limited buffer. To reduce buffer usage, we probabilistically buffer received packet for forwarding. To offset for the loss in performance due to this, we employ network coding[38]. Network coding is a mechanism in which nodes encode two or more incoming packets and forward encoded packets instead of forwarding them as it is.

Fragouli et al. in [39] propose network coding based protocol for all-to-all broadcast in ad hoc network. For network coding, a generation is defined as a group of packets such that packets of the same generation only can be linearly combined to generate encoded packets. Packets from different sources are added in a generation in a distributed

manner. To decide the generation in which a new packet should be added, each source node inspects generations of incoming packets. If the number of packets in a generation (generation size) is less than the threshold, the packet is added in that generation. If no such generations are present then a new generation is created. In typically sparse MSN, packets propagate slowly. As a result, nodes do not receive packets of existing generations quickly enough. So, when the average transmission rate is high, source nodes have to create new generations for their packets as they would not have received information about existing generations in the network. As a result, a large number of generations with small generation sizes are created in the network. It leads to less mixing opportunity and less average delivery ratio as packets of the same generation only can be linearly combined. Widmer et al. in [40] use hashing function for generation management and their technique also has the same problem. Therefore, we propose a community-based distributed generation management technique for MSN to overcome this limitation.

To compare the performance of CAVS with Epidemic routing[3], we modify Epidemic routing such that it delivers packets to all nodes instead of delivering to single destination. Simulation results show that CAVS gives acceptable performance for buffering probability (P_b) value as low as 0.3 as compared to $P_b = 1$. Also, with a decrease in P_b, the performance of CAVS gets increasingly better than Epidemic routing. For $P_b = 0.1$, packet delivery ratio and average packet delivery delay of CAVS is 122% more and 22% less than Epidemic routing respectively. Starting from $P_b = 0.1$, the rate of increase in packet delivery ratio and the rate of decrease in average packet delivery delay in CAVS and Epidemic routing protocols are high till $P_b = 0.3$. So, we recommend operating the protocol with $P_b = 0.3$ approximately. The large performance difference between CAVS and CAVS (without hubs and gateways) for lower P_b values shows that hub and gateway nodes play very significant role in the performance improvement of CAVS protocol.

Results also show that gateway nodes play greater role in the performance improvement of CAVS as compared to hub nodes and improvement in packet delivery ratio and average packet delivery delay beyond 20% of total nodes as popular nodes is not significant. So, we recommend using 20% of total nodes in the network as hub and gateway nodes. Further, packet delivery ratio is almost independent of the maximum generation size (G). But, there is a 12% improvement in average packet delivery delay with $G = 16$ as compared to average packet delivery delay with $G = 2$. Also, after $G = 16$, it remains

almost constant. So, we recommend setting maximum generation size $G = 16$ approximately. With recommended values of buffering probability (P_b), maximum generation size (G), and percentage of total nodes as popular nodes, packet delivery ratio and average packet delivery delay of CAVS is 58% more and 41% less than Epidemic routing respectively.

We model CAVS protocol as SI (Susceptible Infected) epidemic model and show that optimal buffering probability found through simulation is in conformance with the optimal buffering probability calculated using the model.

1.6 Scalable Micro-blogging in MSN

Micro-blogging, particularly Twitter, is very popular among Internet users. Users post small, 140 characters long, messages called tweets. These tweets are about their status, opinions, news, events, emergency situations, traffic updates, advertisements etc. Users tag their tweets based on its content which is called hashtag. Each user receives tweets from users she is following. One can follow a user but not a topic in the system. A user does not get all tweets, with hashtags she is interested in, on her timeline because a huge number of tweets are generated for a hashtag across the Internet. User gets tweets with a hashtag only when she searches for it.

Micro-blogging can be a very promising application for MSN. Small messages posted by users can be opportunistically spread in the network without using any infrastructure. As co-located devices communicate with each other in this network, messages with local and temporal properties can be pushed to users based on their interests without overwhelming them. Users can specify their interests or they can be derived from their posts. Further, co-located devices generally have similar interests[41]. So, users can receive messages in which they are interested in with high probability.

We propose a scalable micro-blogging protocol for MSN which exploits overlapping community structure and heterogeneous popularity of nodes. When two nodes come in contact, each node pushes to the other node messages with tags in which the other node is interested in. To increase chances of delivery to interested nodes, all messages of a community are also forwarded to hub and gateway nodes of the community. Hub nodes help in spreading messages in the community while gateway nodes help in spreading

messages to other communities.

Nodes share their interest profile with community members and hub nodes of the community. Hub and gateway nodes of the network share and accumulate aggregate interest profiles of communities. Based on aggregate interest profiles, the distance threshold is estimated up to which interest is relatively high in the message. Till the distance threshold, messages are forwarded to hub and gateway nodes of the network with probability 1. After the distance threshold, the messages are forwarded to hub and gateway nodes of the network with decreasing probability as distance increases. So, the forwarding overhead of the protocol is limited. Further, nodes do not share their interest profile with nodes of other communities and hub and gateway nodes store only few aggregate entries per tag. As a result, control overhead is also limited. The protocol is scalable as message filtering is done at the network protocol level itself as opposed to the conventional approach where it is done at the end points of the network. It also eliminates the need to maintain a large amount of data at server machines. As server farms consume a huge amount of energy[42], it is useful to not to use infrastructure even though the bandwidth requirement of the application is limited.

Allen et al. in [4] have also proposed a micro-blogging protocol for MSN named as 'Uttering'. We measure and compare 'efficiency' and 'spread index' of our protocol with 'Uttering'. Efficiency is defined as the ratio of total number of useful messages received by all nodes to the total number of messages transmitted by all nodes. Spread index is defined as the ratio of total number of users who have received useful messages to the total number of users who were interested in those messages. Spread index and efficiency measures are same as 'recall' and 'precision' measures used for the performance measurement of 'Uttering' in [4]. Simulation results show that spread index of our protocol is better than existing protocol (Uttering) by 18-59% and efficiency is better than 'Uttering' by 35-56% for different user interest profile models. Simulation results also show that keeping only a constant amount of state information does not degrade the performance of our protocol.

1.7 Contributions of the Ph.D. Work

In this section, we summarize major contributions of the Ph.D. work for efficient peer-to-peer communication in MSN.

We identify that HHW model does not reproduce heterogeneous local popularity of nodes in a community as observed in mobility traces and propose Community Aware Heterogeneous Human Mobility (CAHM) model which rectifies this problem. CAHM also incorporates following properties of human mobility: Levy walk nature of human movement, speed of nodes as a function of distance to be traveled and power-law distributed pause time. We also analyze the effect of mobility models on the performance of routing protocols.

We propose Distributed Overlapping Community Detection (DOCD) mechanism for MSN and also analyze properties of overlapping community structure such as actual community sizes, probability of community existence and fraction of hub and gateway nodes present in the community on an average.

We devise, using mathematical models, methods to identify hub and gateway nodes from the given overlapping community structure itself without doing message flooding or forwarding of accumulated encounter information from other nodes.

We develop a viral spread (many-to-all broadcast) protocol for MSN which exploits properties of human mobility such as overlapping community structure and heterogeneous popularity of nodes for efficient forwarding. The protocol has following salient features: probabilistic buffering of packets at nodes for limited buffer usage, network coding to offset the loss of performance due to probabilistic buffering, and community-based generation management technique for network coding.

We also develop a scalable micro-blogging protocol for MSN which also exploits overlapping community structure and heterogeneous popularity of nodes. The protocol has following salient features: restricted message forwarding based on distance, limited amount of state information which is independent of number of communities and network size and limited control messaging of aggregate interest profiles of communities between hub and gateway nodes only. We also propose three novel models to synthetically generate user interest profiles to analyze the performance of the micro-blogging protocol.

1.8 Organization of the Thesis

In chapter 2, we present a survey of existing mobility models for MSN and propose Community Aware Heterogeneous Human Mobility (CAHM) model. We also analyze the

effect of mobility models on the performance of traditional and socially-aware routing protocols in this chapter. In chapter 3, we survey existing distributed community detection mechanisms for MSN and present Distributed Overlapping Community Detection (DOCD) mechanism for the same. We also analyze properties of overlapping community structure in this chapter. In chapter 4, we discuss existing methods for identifying popular nodes in MSN and describe our methods to identify hub and gateway nodes. In chapter 5, we review related work in the literature for many-to-all broadcast (viral spread) in MSN and present our protocol for viral spread. In chapter 6, we discuss existing protocol for micro-blogging in MSN and propose our protocol for the same. Finally, conclusion and future work are written in chapter 7 and 8 respectively.

Chapter 2

Mobility Modeling for Mobile Social Network

For reliable analysis of the performance of network protocols through simulation, it is very important to model the mobility of communicating devices as realistically as possible. As these devices are carried by humans, their mobility pattern is governed by human mobility pattern. Traditional mobility models, such as Random Way Point (RWP)[16] and Brownian Motion (BM)[17], model device mobility as random. However, researchers have shown that human mobility is rarely random and such models do not provide reliable analysis of network protocol performance[43].

In the following section 2.1, we present literature survey of existing mobility models for Mobile Social Network (MSN). We present our proposed Community Aware Heterogeneous Human Mobility (CAHM) model in section 2.2. In section 2.3, we analyze the effect of mobility models on the performance of routing protocols. We conclude the chapter in section 2.4.

2.1 Survey of Existing Mobility Models for MSN

To study characteristics of human mobility, many experimental studies at various universities (UCSD[44], Dartmouth[45], MIT[13], and University of Illinois[46]) and conferences (Infocom 2005[12], Infocom 2006[7], and SIGCOMM[47]) have been undertaken. In these experiments, humans participating in the experiment carry devices equipped with Wifi/Bluetooth and/or GPS sensor. These devices log encounter, location, and time

information for a period of time.

These real-world mobility traces can directly be used to move nodes in the simulation. But, this approach is not flexible and scalable as the performance of a protocol can be evaluated only for limited values of network parameters for which traces are available. Nonetheless, from analysis of these traces, various statistical properties of human mobility are derived which are as follows.

T.1 Aggregate inter-contact time follows power-law distribution with exponential cutoff[7, 12].

T.2 Pause time follows truncated power-law distribution[14].

T.3 Humans visit nearby locations more frequently compared to far-away locations[15].

T.4 Humans have location preferences and they periodically re-appear at these locations[15].

T.5 Speed at which humans move increases with distance to be traveled[14].

2.1.1 Real-trace Based Models

Real-trace based models try to capture features of individual's independent movement observed from mobility traces. Working Day Mobility (WDM) model[48] incorporates properties T.1 and T.4 by modeling individual's mobility during a day with home sub-model, office sub-model, transport sub-model, and evening sub-model. Time Variant Community (TVC) model[18] also incorporates properties T.1 and T.4. Small World In Motion (SWIM) model[19] incorporates all properties T.1 to T.5. In this model, each node is assigned a randomly and uniformly chosen point over the network area called home. For each node, a weight is assigned to each possible destination which grows with the popularity of the place and decreases with distance from home. This weight represents the probability for the node to choose that place as next destination. Self-similar Least Action Walk (SLAW) model[20] incorporates properties T.1, T.2 and T.3.

2.1.2 Social-aware Models

Although real-trace based models are able to reproduce statistical properties of mobility traces, they assume that each node moves independently. But, movement of an individual

is not independent. Humans belong to multiple social communities like friends, family, co-workers, etc.[9, 10]. These social ties significantly affect their movement. Following are the main characteristics derived from the social network theory which affect human mobility.

S.1 Humans form communities based on their social relationships[9].

S.2 Humans belong to multiple communities and so, communities overlap[10].

S.3 Different individuals have different local popularity within a community and different global popularity in the social network[5].

S.4 Community size, the number of communities in which a node is a member and overlap size approximately follow power-law distribution where overlap size is defined as the number of individuals which are common in two communities[10].

Community-based Mobility Model (CMM)[11] groups nodes based on social relationships among individuals carrying these nodes. This grouping is then mapped to a topographical space. Movement of nodes is influenced by the strength of social ties among individuals which may also change in time. CMM uses Caveman model[22] as artificial Social Network Model (SNM) to generate community structure. Home-cell Community-based Mobility Model (HCMM)[21] assigns the home cell to each individual which is the location where people with whom the node shares social relationships are likely to be at some point in time. After each trip, the node moves to the home cell with some probability. N-body[49] learns the grouping behaviour of a small set of nodes from their mobility traces and generates mobility traces for a large number of nodes exhibiting similar grouping behaviour. These models incorporate only S.1 of social network theory based properties. CMM and HCMM also incorporate some of the properties derived from mobility traces. But, these models do not incorporate properties S.2, S.3, and S.4 which are very important properties and have a significant effect on the performance of routing protocols. Social, sPatial, and Temporal mobility framework (SPoT)[50] is flexible and controllable mobility framework. But, it generates only contact traces and proposal in the paper for generating movement traces is preliminary. Further, it takes a social graph as an input instead of generating community structure synthetically. So, it lacks the flex-

Table 2.1: Comparison of mobility models for MSN

Mobility model	T.1	T.2	T.3	T.4	T.5	S.1	S.2	S.3	S.4
SLAW[20]	✓	✓	✓						
WDM[48]	✓	✓	✓	✓					
TVC[18]	✓	✓	✓	✓					
SWIM[19]	✓	✓	✓	✓	✓				
N-body[49]	✓					✓			
CMM[11]	✓			✓		✓			
HCMM[21]	✓		✓	✓		✓			
HHW[1]	✓			✓		✓	✓	✓	✓
CAHM (Proposed)	✓	✓	✓	✓	✓	✓	✓	✓	✓

ibility of generating a large number of different social graphs for simulation. A detailed review of human mobility in opportunistic networks is available in [51].

Heterogeneous Human Walk (HHW) model[1] incorporates all properties S.1 to S.4 derived from social network theory to generate community structure synthetically. HHW is able to generate any number of overlapping community structures on its own based on input parameters. It does not take real life social network as an input, as requiring real life social network as an input restricts possible scenarios for which performance evaluation can be done. Further, it also does not use Social Network Models (SNM) such as Caveman model[22] to generate community structure, as these models are quite simplistic and do not take into account all social network theory based properties. However, HHW does not incorporate following important trace-based properties: i) Levy walk nature of human mobility ii) speed of nodes as a function of distance to be traveled and iii) power-law distributed pause time. Further, we show that HHW model does not correctly reproduce the heterogeneous local popularity of nodes in a community as observed in mobility traces. So, we propose Community Aware Heterogeneous Human Mobility (CAHM) model which incorporates these properties and also generates correct heterogeneous local popularity. We summarize the comparison of different mobility models for MSN in Table 2.1.

As CAHM is based on HHW, we give an overview of HHW model in the following sub-section.

2.1.3 Overview of Heterogeneous Human Walk (HHW) Mobility Model

There are two options to generate overlapping community structure. First is to get a social graph from some real life social network. After getting the social graph, one can apply algorithm similar to the one proposed in [10] to identify overlapping communities. But, it will significantly increase implementation and computational complexity and one has to collect a large number of social graphs to analyze the performance of protocols. It is also less flexible and scalable. The second option is to directly construct synthetic overlapping community structure which follows all the properties found in real life social networks. To achieve a trade-off between reality and complexity, HHW uses the second approach. In this approach, any number of different community structures can be generated using different random seeds.

In overlapping community structure, each individual n in the social network may belong to number of communities denoted as membership number Λ_n. Further, any two communities x and y may share $S^{ov}_{x,y}$ individuals, defined as overlap size between two communities. Let us denote size of community x as S^{com}_x and probability distribution functions of membership number, overlap size and community size as $\mathrm{P}(\Lambda)$, $\mathrm{P}(S^{ov})$ and $\mathrm{P}(S'^{com})$ respectively. Here, $S'^{com} = S^{com} - k$ to keep minimum community size equal to k where k is clique size. A k-clique is complete sub-graph of size k and k-clique community is union of all k-cliques that can be reached from one another through series of adjacent k-cliques where two k-cliques are adjacent if they share $k-1$ nodes[10]. Based on the analysis of a variety of social networks, Palla et al.[10] conclude that $\mathrm{P}(\Lambda)$, $\mathrm{P}(S^{ov})$ and $\mathrm{P}(S'^{com})$ approximately follow power-law distribution $\mathrm{P}(x) \sim x^{-\tau}$, with exponents $\tau = \Upsilon_\Lambda$, $\tau = \Upsilon_{Osize}$ and $\tau = \Upsilon_{Csize}$, respectively. Further, they report that values of Υ_Λ and Υ_{Osize} are not less than 2, and the value of Υ_{Csize} is between 1 and 1.6. HHW model uses these statistical properties to synthetically construct k-clique overlapping community structure.

HHW model is composed of four components: 1) Establishing overlapping community structure, 2) Generating heterogeneous local degree, 3) Mapping communities into geographical zones, and 4) Driving individual motion. These components are explained in following four sub-sections.

Establishing k-clique Overlapping Community Structure

A day (or a week or any time duration) is divided into periods, and overlapping community structures are different in each of these periods but is same in the same period of different days. Let us define nodes with membership number larger than 2, equal to 2 and equal to 1 as M-3 nodes, M-2 nodes, and M-1 nodes respectively. Community structure for each period is constructed as follows:

1. Generate nodes' membership numbers such that they follow P(Λ) with exponent Υ_{Λ}. Then, establish initial empty communities whose sizes S^{com} follow P($S^{'com}$) with exponent Υ_{Csize} such that $\sum_i \Lambda_i = \sum_j {S_j}^{\text{com}}$.

2. Use all M-3 nodes to establish initial overlaps between pairs of communities.

3. Modify initial overlaps by allocating all M-2 nodes to communities such that overlaps' sizes follow P(S^{ov}) with exponent Υ_{Osize}.

4. Allocate all M-1 nodes to unsaturated communities.

Generating Heterogeneous Local Degree

Local degree of a node within a community is defined as the number of neighbours of the node in the community. A node's local popularity depends on its local degree. Let $Local_i^n$ denote local degree of node n in its community i where $Local_i^n \geq k-1$ as per the definition of k-clique community. These values are generated such that they follow a power-law distribution with exponent Υ_{Local}.

Mapping Communities into Geographical Zones

To simulate n mobile nodes in a two-dimensional square plane, the model divides the plane into a grid of non-overlapping square cells. For each period, each community with size S^{com} is associated with a zone composed of S^{com} adjacent cells. The location of a zone within the simulation plane is chosen randomly such that zones of different communities do not overlap. Each node n is randomly associated with $Local_i^n$ cells within the zone of its community i.

Driving Individual Motion

Initially, each node randomly selects one of its associated cells and then it is located at a random position inside that cell. To move, each node selects next goal inside one of its associated cells and then moves towards it by following the straight path. The speed at which it moves towards next goal is chosen randomly with uniform distribution. When a node reaches its current goal, it waits for a uniformly distributed pause time and then selects and moves towards next goal. The overlapping community structure, corresponding associated zones and cells change at the start of the new period. When the period changes, after reaching its current goal, the node selects next goal inside one of its newly associated cells of the new period.

The paper[1] demonstrates that nodes of this overlapping community structure exhibit global heterogeneous popularity and aggregate inter-contact time follows power-law distribution with exponential cutoff.

2.2 Community Aware Heterogeneous Human Mobility (CAHM) Model

In this section, we discuss shortcomings of existing HHW model and propose our solution for each of them.

2.2.1 Incorporating Levy Walk Nature of Human Mobility

In HHW, if a node is a member of more than one community, then it will have associated cells in all those communities. As per the model, the node chooses an associated cell as next goal randomly with uniform distribution irrespective of the community of the associated cell; i.e. the node chooses next goal irrespective of the distance it will have to travel. This is contrary to the finding that humans prefer short distances over long distances or, in other words, human movement can be characterized as Levy walk[14]. It is also counter-intuitive, e.g. a postman has to visit multiple offices in multiple buildings to deliver posts. As per the existing model, the postman will move from an office in a building to another office in another building with more probability than to the office in the same building assuming that the total number of offices of other buildings is more

than the number of offices in the same building.

As established in [14], distances covered in flights by a person follow a power-law distribution with exponent less than 2.5 where flight can be defined as single displacement from one place to another place without a pause in between. We have incorporated this property in our model.

In CAHM, a node chooses an associated cell as next goal based on the distance it will have to travel with truncated power-law distribution instead of choosing it randomly with uniform distribution. We calculate distances at which all associated cells of a node are located from the current cell from their location information and sort associated cells of a node based on these distances. Then, we use random variate (RV) following power-law distribution P(D) with exponent Υ_D between the minimum distance and the maximum distance a node has to travel. We choose the associated cell as next goal whose distance from the current cell is nearest to the value generated by the random variable. As RV will generate short distance values with higher probability than long distance values, in our model, a node prefers short distances over long distances. As each community is associated with a zone composed of adjacent cells, distance between any two cells within a community will most probably be less than distance between any two cells of two different communities if the simulation area is not too small. Therefore, a node will be choosing one of the associated cells of the current community as its next goal with high probability compared to the associated cells of the node in other communities, which is correct behaviour as we have seen in the postman example.

2.2.2 Generating Correct Heterogeneous Local Popularity

In HHW, each community with size S^{com} is associated with a zone composed of S^{com} adjacent cells; i.e. number of nodes (N) and number of cells (C) of a community are same. So, the average number of nodes associated with a cell is equal to the average local degree of a community (μ). One important effect of it is that a node, on an average, meets μ number of nodes in each cell it visits. So, the local popularity of nodes increases with an increase in local degree at the rate proportional to μ times rate of increase in local degree instead of increasing with the same rate. As a result, there are a higher percentage of nodes with high local popularity than observed in mobility traces[5]; i.e. HHW generates

too many locally popular nodes than expected. Further, μ increases with an increase in community size because local degree follows truncated power-law distribution with the maximum value equal to community size. So, the problem is aggravated in large communities.

In CAHM, we consider C and N as separate parameters. Let C_x be the number of adjacent cells, μ_x be the average local degree and N_x be the number of nodes in community x. Let m be the community density index denoting denseness of a community. Then,

$$C_x = m \times \mu_x \times N_x \tag{2.1}$$

2.2.3 Calculating Speed Based on Distance

In HHW, the speed at which a node moves from one goal to next goal is chosen from given range uniformly regardless of the distance to be traveled. But, as found in [14], speed increases with the increase in flight length because individuals use transportation to travel long distances instead of walking. They have also derived following relation between flight time (t) and flight length (l) from different mobility traces.

$$t = p \times l^{1-\eta}, 0 \leq \eta \leq 1 \tag{2.2}$$

From mobility traces, Rhee et al.[14] have proposed $p = 30.55$ and $\eta = 0.89$ when $l < 500$ meters, and $p = 0.76$ and $\eta = 0.28$ when $l \geq 500$ meters. In CAHM also, we use this model to calculate speed at which a node should travel to next goal instead of choosing it uniformly from given range.

2.2.4 Simulation Results

We have implemented HHW and CAHM model in ONE simulator[52]. It is a de facto simulator for Delay Tolerant Network (DTN) research. We simulate HHW and CAHM models with following scenario. There are 200 nodes in a simulation plane of 5000 meters x 5000 meters, divided into a grid of 62,500 cells of 20 meters x 20 meters each. The transmission range of each node is 20 meters. For HHW, the speed of a node is uniformly distributed between 1 and 6 m/s. For CAHM, speed follows Eq. 2.2 and pause time is

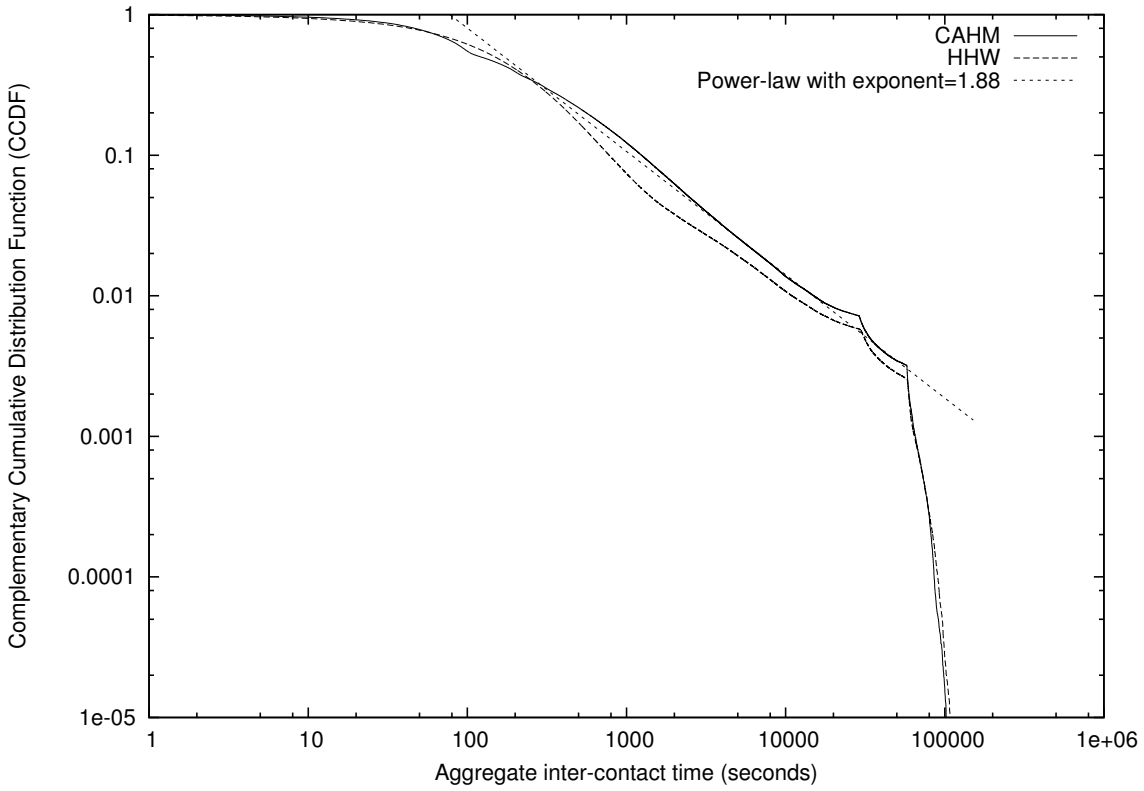

Figure 2.1: Complementary Cumulative Distribution Function (CCDF) of aggregate inter-contact times

generated using power-law distribution with exponent 2 between 0 and 1000 seconds. We generate 4-clique communities, i.e. with $k = 4$. We set $\Upsilon_{\Lambda} = 3$, $\Upsilon_{Osize} = 2$, $\Upsilon_{Csize} = 1.2$, $\Upsilon_{Local} = 2.5$, and flight length exponent $\Upsilon_{D} = 2$. All these values are in the range recommended for these exponents in the literature from mobility traces[14, 9, 10]. For comparison, we use same community structures for both HHW and CAHM model.

To verify that in CAHM also, aggregate inter-contact time distribution is power-law with exponential cutoff, we simulate HHW and CAHM model for two days and each day is divided into three identical periods of 8 hours each. We generate three different overlapping community structures for each period using random variables which follow power-law distribution with exponents for different quantities as specified earlier. As shown in Fig. 2.1, Complementary Cumulative Distribution Function (CCDF) of aggregate inter-contact times of CAHM follows power-law distribution with exponential cutoff. It matches with CCDF of aggregate inter-contact times of HHW which is shown to be matching with CCDF of aggregate inter-contact times of mobility traces[1].

To understand the behaviour of nodes with membership in multiple communities, we

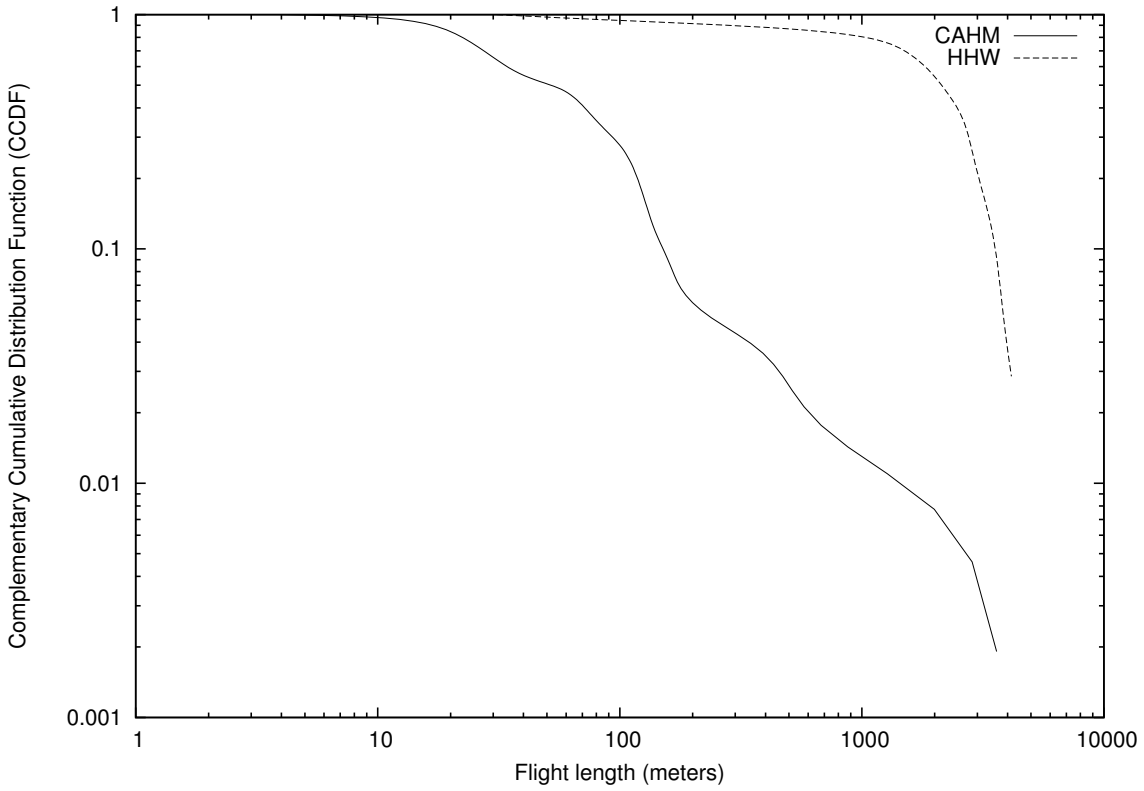

Figure 2.2: Complementary Cumulative Distribution Function (CCDF) of flight lengths

select a node which is a member of 11 different communities and record flight lengths of all its flights both in HHW and CAHM. As shown in Fig. 2.2, flight lengths in CAHM follow power-law distribution while, in HHW, flight lengths do not follow power-law distribution.

As shown in Fig. 2.3, in HHW, the node takes high number of long flights; i.e. it jumps from one community to another community frequently while, in CAHM, the node moves within one community most of the time before jumping to another community.

In the simulation, in order to measure local popularity, we count the number of encounters between all pairs of nodes in a community. Let, for a pair of nodes, the average number of such encounters be μ_e and variance be σ_e. Then, the node's local neighbours are the nodes in the community it has encountered more number of times than $\mu_e + \sigma_e$. Number of such local neighbours denotes the node's local popularity.

Fig. 2.4 shows CCDF of local popularities of nodes in a community of size 70 in HHW and CAHM with community density index m=1, 2, and 3 (Eq. 2.1). As conjectured, it is evident from the figure that in HHW, there are too many nodes with high local popularity than expected while CAHM generates power-law distributed local popularity. Further, in CAHM, with an increase in the value of m (i.e. with a decrease in denseness)

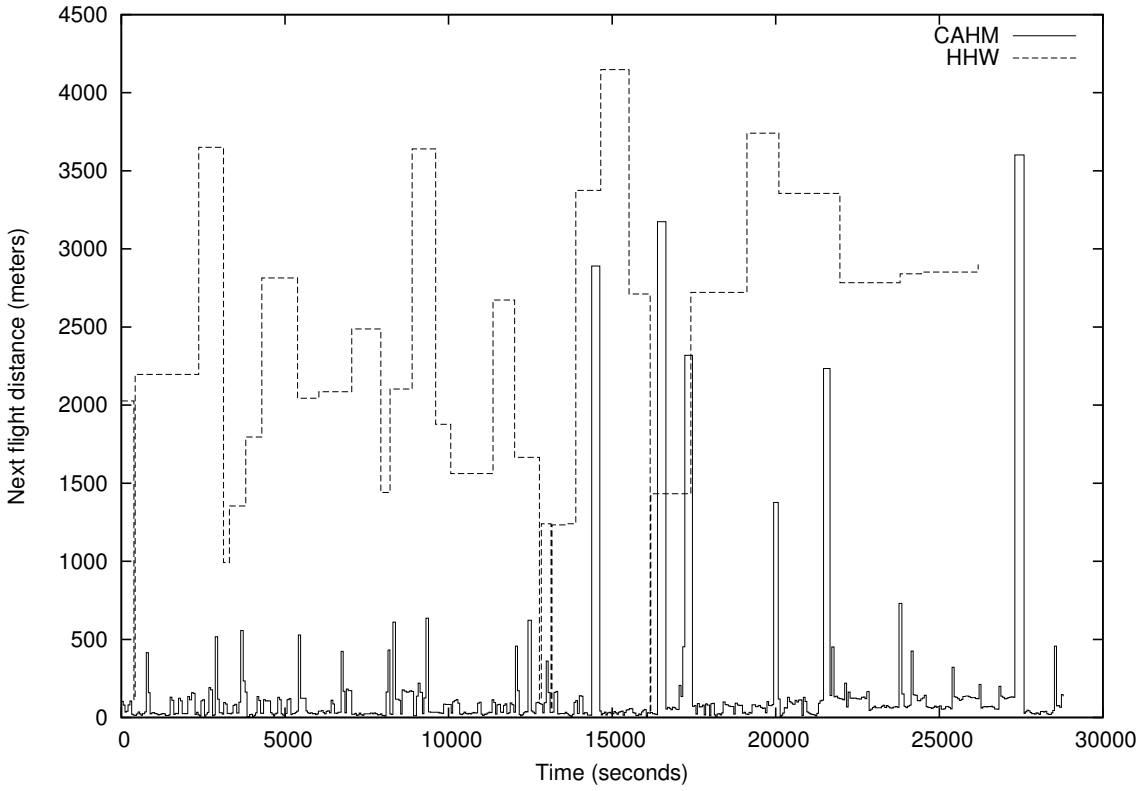

Figure 2.3: Next flight distance v/s Time (Large distance implies inter-community transition)

local popularity of nodes decreases. So, by changing the value of m, one can control the local popularity of nodes in a community.

2.3 Effect of Mobility Models on the Performance of Routing Protocols

In order to analyze effect of mobility models on the performance of routing protocols, we compare packet delivery ratio and delay for Epidemic routing[3] and BUBBLE Rap[5] protocols with CAHM, HHW, CMM, and RWP mobility models. HHW and CMM incorporate some of the properties of human mobility listed in section 2.1 while RWP does not incorporate any of these properties. Epidemic routing does not exploit any of the properties of human mobility while BUBBLE Rap exploits community structure and heterogeneous popularity of nodes to achieve better performance.

Implementation of Epidemic routing is available in ONE simulator. For CMM, Musolesi et al.[11] have provided an implementation which generates mobility scenario for

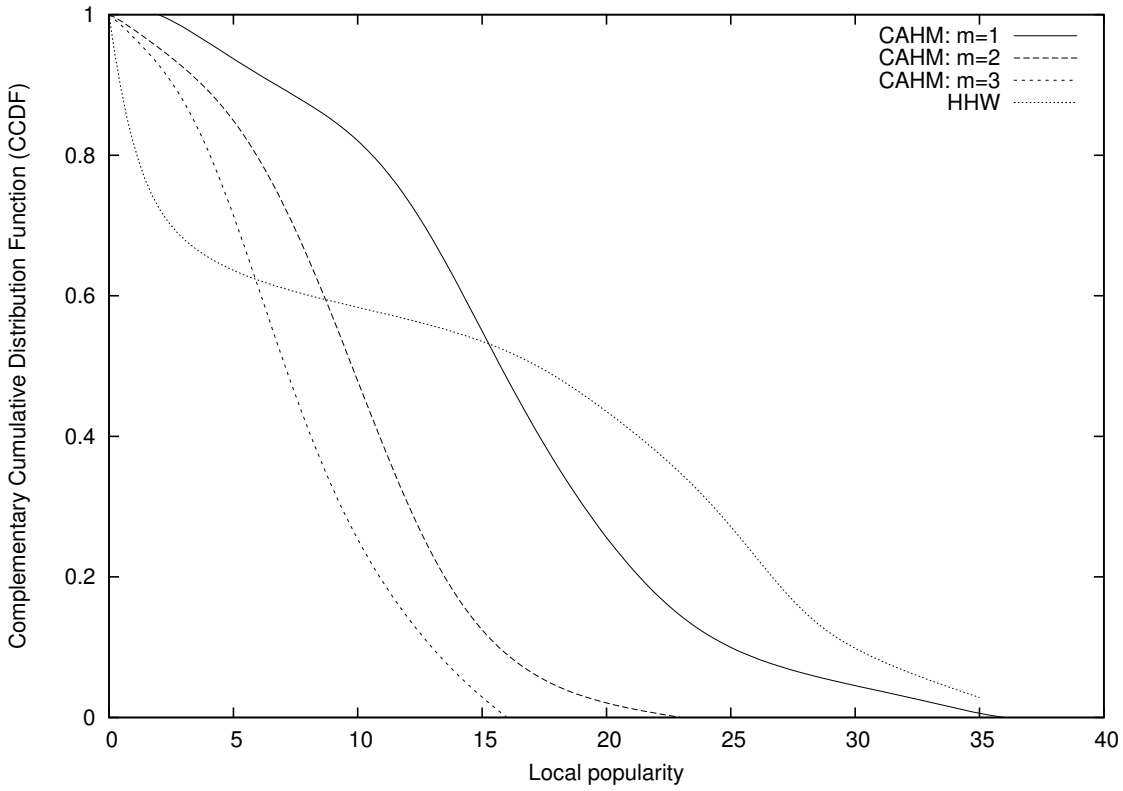

Figure 2.4: Complementary Cumulative Distribution Function (CCDF) of local popularities

Network Simulator (NS-2). We have implemented conversion class in ONE simulator which moves nodes in ONE simulator as per the mobility scenario generated for NS-2.

We move 200 nodes in the area of 5000 meters x 5000 meters. In CAHM, the speed of individual flight is calculated based on Eq. 2.2. The equation and values of terms in the equation are derived in [14] from mobility traces of four different places. For our simulation scenario, with CAHM, average flight speed turns out to be 5.43 m/s. So, we set average flight speed as 5.43 m/s with uniform distribution in the range [0.09, 10.77] for other mobility models. In CAHM, the pause time is power-law distributed with exponent 2.39, minimum pause time 25 seconds, and maximum pause time 9500 seconds. These values are derived in [14] from mobility traces of a university campus referred as 'Campus I' in the paper. For our simulation scenario, with CAHM, average pause time turns out to be 83.82 seconds. So, we set average pause time as 83.82 seconds with uniform distribution in the range [25, 142.64] for other mobility models. In CAHM, the flight length exponent is 1.29. The transmission speed of nodes is 3 MBps and transmission range is 20 meters. Nodes are not buffer constrained. Packets are of 8 KB size. We vary inter-packet arrival

time from 1 to 8 seconds.

For HHW, CAHM, and CMM, the number of communities are 13. For HHW and CAHM, cell size is 100 meters x 100 meters and remaining parameters are as specified in sub-section 2.2.4. For CMM, the number of rows and columns are 7. It is chosen such that node density within a community remains same in HHW, CAHM, and CMM. Re-wiring probability for CMM is 0.1.

It is evident from Fig. 2.5 that packet delivery ratio of both Epidemic routing and BUBBLE Rap routing is significantly greater with CAHM mobility model as compared to other three mobility models. As packet delivery ratio of BUBBLE Rap with CAHM is much higher than packet delivery ratio with HHW and CMM, it can be said that packet delivery ratio increases significantly with CAHM because of heterogeneous local popularity of nodes as observed in mobility traces and also because of following additional properties of human mobility which are not incorporated in HHW and CMM but are part of CAHM: Levy walk nature of human mobility, speed of nodes as a function of distance to be traveled, and power-law pause time.

It is also evident from Fig. 2.5 that the packet delivery ratio of Epidemic routing is greater than BUBBLE Rap as it does around 50% more transmissions per packet than BUBBLE Rap. But, the difference in the packet delivery ratio of Epidemic routing and BUBBLE Rap with CAHM, HHW and CMM models is less pronounced as compared to with RWP. As seen in Fig. 2.6, similar results are obtained for average packet delivery delay also.

To check whether the results are similar for different node density, we repeat the simulation in the 13000 m x 13000 m area. For this area, with CAHM, average flight speed turns out to be 8.23 m/s. So, we set average flight speed as 8.23 m/s with uniform distribution in the range [0.35, 16.11] for other mobility models. Further, with CAHM, average pause time turns out to be 82.32 seconds. So, we set average pause time as 82.5 seconds with uniform distribution in the range [25, 140] for other mobility models. With these parameters too, we get similar results. The results confirm that exploiting community structure and heterogeneous node popularity significantly improves performance.

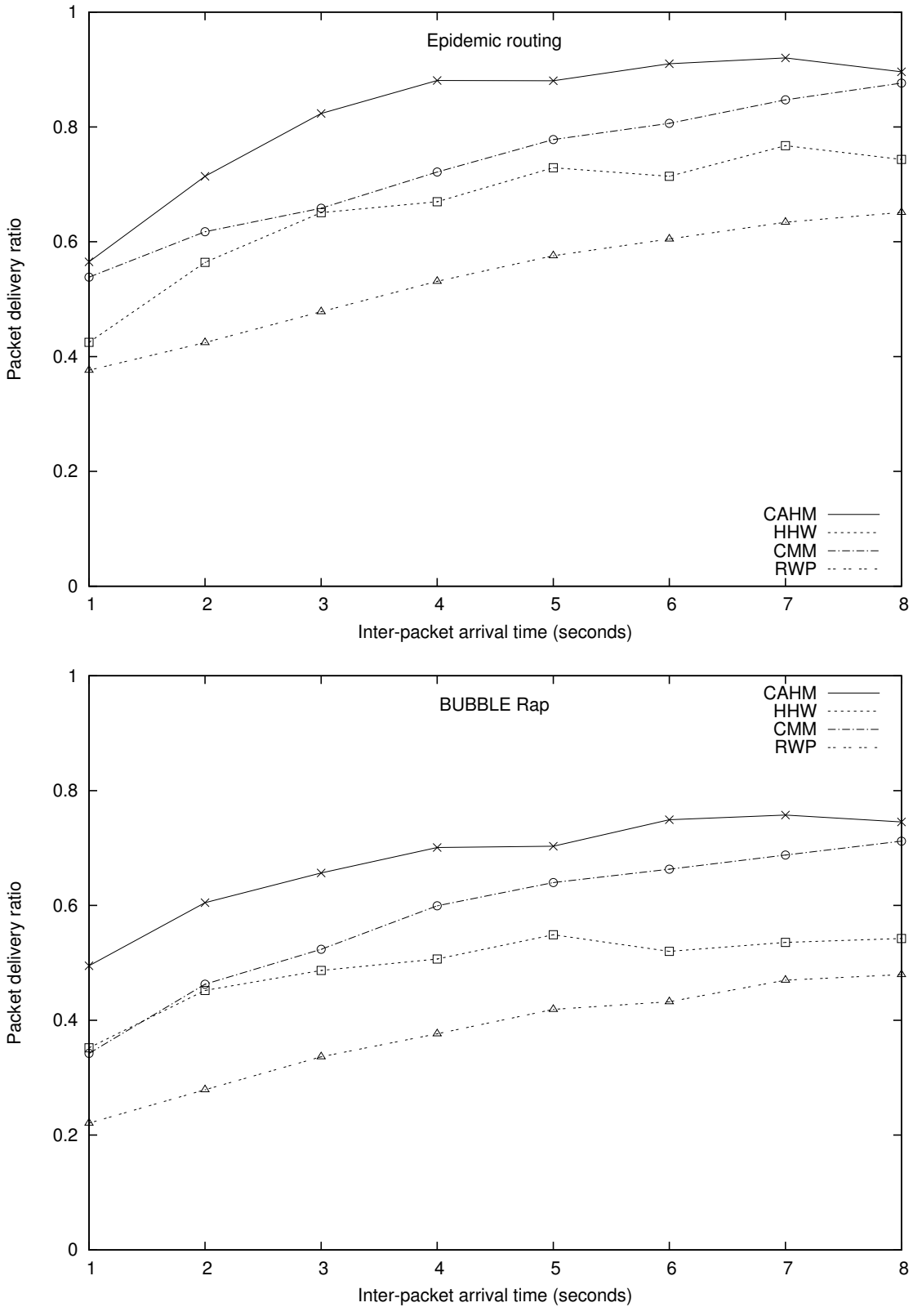

Figure 2.5: Packet delivery ratio v/s Inter-packet arrival time

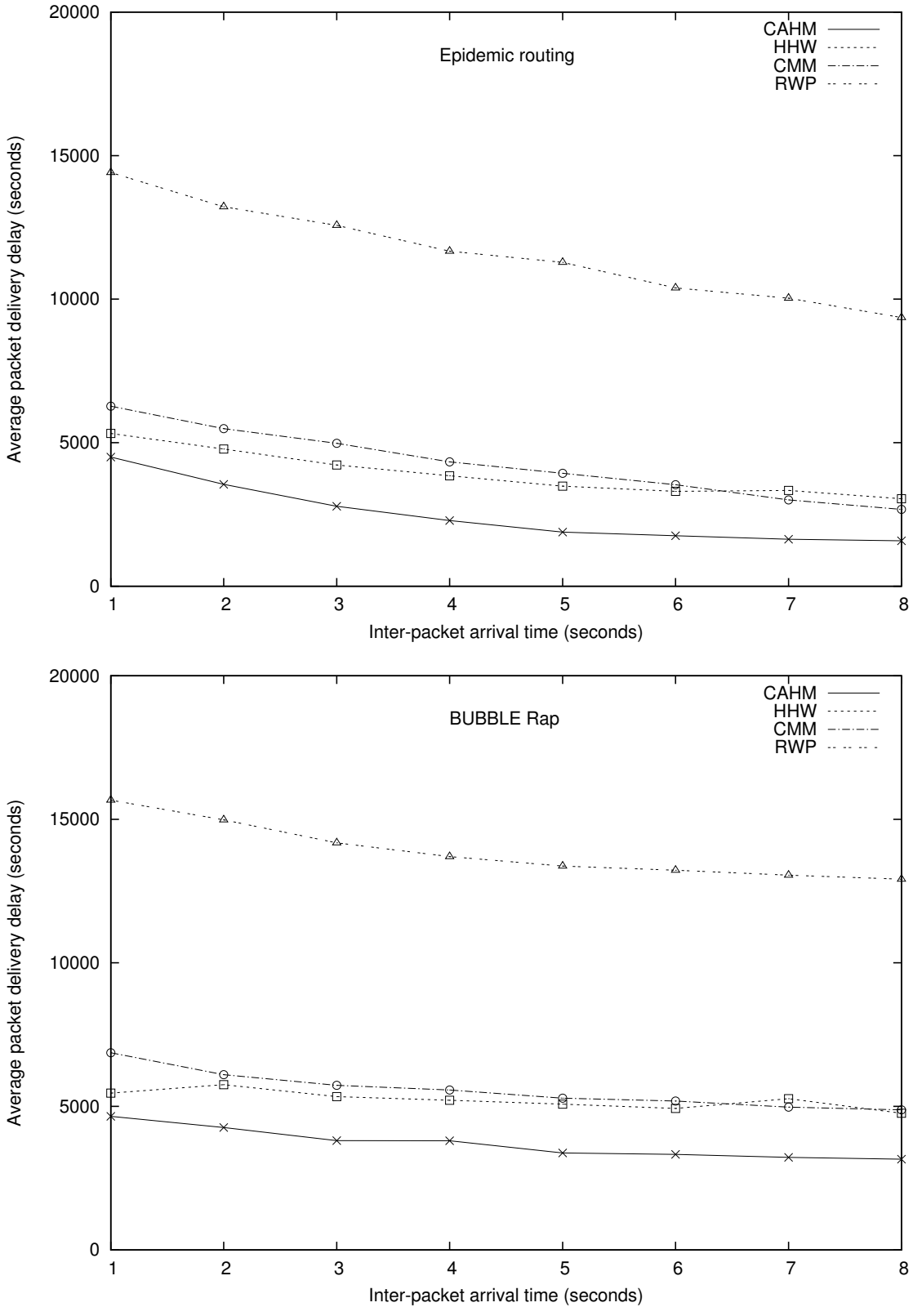

Figure 2.6: Average packet delivery delay v/s Inter-packet arrival time

2.4 Conclusion

Several real-world mobility traces have established that human mobility is not random. So, traditional mobility models such as Random Way Point (RWP) and Brownian Motion (BM) should not be used for reliable analysis of protocol performance in MSN through simulation. Various characteristics of human mobility are derived from mobility traces and from social network theory in the literature. No existing scalable and flexible mobility model incorporates all these characteristics.

We propose Community Aware Heterogeneous Human Mobility (CAHM) model with four modifications in HHW: incorporation of Levy walk nature of human mobility, treatment of number of cells and number of nodes in a community as separate parameters, calculation of speed based on distance to be traveled and power-law pause time. Simulation results demonstrate that CAHM successfully generates flight lengths with power-law distribution while in HHW flight lengths are uniformly distributed. Further, movement of individuals in CAHM is as per rational human behaviour of preference of nearby locations over far-away locations while in HHW it is not. The Results also establish that CAHM generates desired heterogeneous local popularity of nodes while HHW generates too many highly popular nodes.

We analyze the effect of mobility models on the performance of routing protocols in MSN. Simulation results confirm that exploiting community structure and heterogeneous node popularity significantly improves performance. The results also show that due to less realistic mobility models, simulation significantly underestimates the performance of protocols. The packet delivery ratio of Epidemic routing and BUBBLE Rap routing is 21-28% higher with CAHM as compared to HHW in our simulation setup. The result shows that additional properties of human mobility incorporated in CAHM such as heterogeneous popularity of nodes as observed in mobility traces, Levy walk nature of human mobility, speed as a function of distance, and power-law pause time have significant impact on the performance of routing protocols.

Chapter 3

Distributed Overlapping Community Detection in Mobile Social Network

Humans belong to multiple communities, i.e. communities in social network overlap[10]. These social ties significantly affect humans' movement pattern and, in turn, contact pattern of mobile nodes carried by them. Social-aware routing protocols are designed to exploit this contact pattern for efficient communication in Mobile Social Network (MSN). Such routing protocols can be categorized as i) explicit community detection based and 2) implicit community detection based protocols. Explicit community detection based protocols[27, 29, 28, 5, 30, 31] detect underlying community structure of MSN and use this structure for forwarding decisions. Implicit community detection based protocols[53, 54, 55, 56, 57, 58, 59] do not detect underlying community structure explicitly but uses different metrics to capture human social behaviour.

It is desirable that the community detection mechanism employed by explicit community detection based protocols is distributed in nature because of the peer-to-peer nature of MSN. Further, as humans belong to multiple communities, community detection mechanism should be able to find overlapping communities. In section 3.1, we survey distributed community detection mechanisms for MSN available in the literature and conclude that there is no distributed overlapping community detection mechanism available in the literature. So, we propose Distributed Overlapping Community Detection (DOCD) mechanism for MSN in section 3.2. We also analyze properties of overlapping community structure in section 3.3. We conclude the chapter in section 3.4.

3.1 Survey of Distributed Community Detection in MSN

In MSN, some nodes come in contact with each other more frequently or remain in contact for a longer duration than other nodes in the network. This contact pattern can be represented as a graph. Vertices in the graph represent nodes. An edge between two vertices is added in the graph if, for corresponding nodes in MSN, average inter-contact time is less than some threshold or total contact duration is greater than some threshold. As humans form communities, the graph created from MSN is not a random graph. The degree distribution of the graph often follows a power-law distribution, i.e. there are many vertices with a low degree and some vertices with a high degree. Further, distribution of edges is also inhomogeneous with a high number of edges within groups of vertices and a low number of edges between these groups. These groups are called communities.

Community detection in graphs is a well-studied topic as any network can be represented as a graph. The problem of community detection in graphs has applications wherever something can be represented as a network and there is a requirement of detecting groups (communities) in the network; e.g. online social networks, protein-protein interaction networks, World Wide Web (WWW) etc. A very good and detailed review for community detection in graphs is presented in [60]. But, all the techniques discussed in the paper are centralized with the assumption that the entire graph of the network is available at one place. So, these techniques are not directly applicable for MSN where the requirement is to detect communities in a distributed manner at each node from the contact pattern available locally.

Hui et al. in [2] have proposed three algorithms for distributed community detection in Delay Tolerant Networks (DTN) namely MODULARITY, K-CLIQUE, and SIMPLE. These algorithms are widely used by a number of explicit community detection based routing protocols for MSN. These algorithms differ in the level of complexity and overhead. MODULARITY is most complex with highest overhead while SIMPLE is least complex with least overhead. The accuracy of communities detected by these algorithms as compared to centralized mechanism varies with MODULARITY algorithm being most accurate. But, the difference in accuracy of these algorithms is not very significant as authors report the accuracy of all algorithms between 80% and 90%.

In these algorithms, each node detects and maintains its own single local community. Each node in the network maintains cumulative contact duration information for each node with which it comes into contact. If encountered node's cumulative contact duration is greater than given threshold then that node is added to the familiar set as well as to the local community of the node (*threshold criteria*). When a node encounters another node, it exchanges its local knowledge of the network with its neighbour. Then, each node decides whether to add encountered node in its local community based on *admission criteria* if it is not already added based on *threshold criteria*. It also decides whether to merge encountered node's local community with its local community based on *merging criteria*. All three algorithms differ in the *admission criteria* and *merging criteria* while *threshold criteria* is common for all the algorithms.

The problem with these algorithms is that there is no aging mechanism involved. Once a node is added to the local community of some node, it is never dropped from the local community even if it is not encountered for a long period of time. The AD-SIMPLE[23] algorithm modifies SIMPLE and adds an aging mechanism to rectify this problem. It prunes a node from the familiar set of the current node if the running average of the percentage of contact duration of the node in a given time period falls below a given threshold. It also prunes a node from the local community of the current node if, in a given time period, the node is encountered neither directly by the current node nor by any community member of the current node. If a node is pruned from the local community and it is present in the familiar set then it is also removed from the familiar set. Orlinski et al. in [24] propose a distributed mechanism to detect communities which change over space and time. It also employs aging mechanism similar to AD-SIMPLE to account for the changes in community membership over space and time.

Another problem with MODULARITY, k-CLIQUE, SIMPLE, AD-SIMPLE, and the mechanism in [24] is that they do not detect multiple communities of a node. We know that nodes in MSN can belong to multiple communities like friends, family, co-workers etc. But, MODULARITY, k-CLIQUE and SIMPLE merge all these communities and detect single community per node. AD-SIMPLE does not merge these communities but maintains only one of these communities as the local community of the node at any given point of time. It is desirable for a node to maintain information about all the communities in which it is a member. This information can be used by routing protocols

to make better forwarding decisions. Further, as we shall see in chapter 4, we also use this information to detect hub and gateway nodes of communities without doing message flooding or forwarding of accumulated encounter information from other nodes.

Williams et al. in [25] propose a mechanism for decentralized detection of periodic encounter communities in opportunistic networks. Periodic encounter community means that a community is formed periodically when members of the community come in contact with each other. Then, it gets dissolved as its members move away from each other; e.g. during office hours, a community is formed every day between office workers which dissolves after office hours. So, this mechanism can detect different communities in which a node is a member in different periods. But, it is evident from the example discussed in 1.1, a node can be part of different communities within any given period and nodes can move between communities without communities getting dissolved. We call such a community structure as an overlapping community structure.

So, we propose Distributed Overlapping Community Detection (DOCD) mechanism by modifying SIMPLE algorithm to detect multiple communities of each node in the following section 3.2.

3.2 Distributed Overlapping Community Detection (DOCD) in MSN

Out of MODULARITY, k-CLIQUE, and SIMPLE algorithms for distributed community detection, SIMPLE is the least complex and control message overhead of SIMPLE is also minimum as compared to MODULARITY and k-CLIQUE. Further, accuracies of these algorithms are similar. So, we choose to modify SIMPLE to detect multiple communities. As such, our mechanism can work with any of these algorithms.

As our mechanism (DOCD) is based on SIMPLE, in the next sub-section 3.2.1, we give an overview of the SIMPLE algorithm. Detailed description of the same can be found in the original paper[2]. In sub-section 3.2.2, we discuss our mechanism to detect multiple communities. We propose two measures to evaluate the performance of any distributed overlapping community detection mechanism in sub-section 3.2.5. Simulation results for DOCD are discussed in sub-section 3.2.6.

3.2.1 SIMPLE Algorithm for Distributed Community Detection

As discussed before, familiar set (F_v) of a node v is set of all encountered nodes for which cumulative contact duration is greater than given familiarity threshold. It is called *threshold criteria*. Local community (C_v) of a node v includes all the nodes in its familiar set and the nodes added through *admission criteria* and *merging criteria*. When two nodes comes into contact with each other, each node sends the other node its familiar set and the local community. Using this information, each node updates its local community based on following two criteria. Let the nodes be v_i and v_j. Then, node v_i updates its local community C_i as follows.

- *Admission criteria*: If *threshold criteria* is not satisfied for node v_j then *admission criteria* is applied. If number of nodes shared among C_i and F_j is higher than λ times the number of nodes in F_j then the encountered node v_j is added in the local community (C_i) of v_i. Here, $0 < \lambda \leq 1$ is a parameter of the algorithm. Intuitively, a node is added in a community if majority nodes in the familiar set of the node are also in the community.

- *Merging criteria*: If *admission criteria* is satisfied then *merging criteria* is applied. If the number of nodes shared between C_i and C_j is higher than γ times the number of nodes in the set union of C_i and C_j then all members of C_j are added in C_i also. Here, $0 < \gamma \leq 1$ is another parameter of the algorithm. Intuitively, two communities are merged if they share significant number of nodes as their members.

To detect multiple communities, we add following proposed mechanism in the SIMPLE algorithm.

3.2.2 The Proposed Mechanism to Detect Multiple Communities

As different communities are associated with different locations in MSN, the mechanism identifies a community by its location. For the same, a node maintains a list of locations it has visited in a community. The node identifies the community with the centroid of these locations.

3.2.3 Recording Visited Locations

On each contact with another node, each node updates its current community as per SIMPLE algorithm. It also adds *contact location* as a new location it has visited in the current community if the distance between *contact location* and other locations it has visited in the current community is greater than some threshold. Thus, each node maintains a separate list of locations it visits in a community for each detected community.

Centroid of a Community

Each node identifies its detected communities by the centroid of these communities. For a node, the centroid of a community is the geometric center of all the locations of the community the node has recorded. The node updates the centroid of a community whenever the list of locations it visits in a community is updated. As locations of a community visited by member nodes of the community will be different from each other, each node will have different centroid for the community.

Adaptive Threshold Distance (d_{th})

At a node, threshold distance ($d_{th}^{C_i}$) of community C_i is calculated as the distance between the centroid of the community and the farthest location of the community visited by the node. Initially, its value is set as an input parameter to the mechanism and it is common for all nodes. Then, each node maintains different threshold distance values for each detected community.

Packet forwarding protocols can check whether two communities C_i and C_j detected by two different nodes are same or not by calculating the distance between two corresponding centroids. If the distance is less than minimum of threshold distances $d_{th}^{C_i}$ and $d_{th}^{C_j}$ then C_i and C_j are considered to be the same.

Recording Detected Community

On each contact with another node, a node checks for the distance it has traveled from the centroid of the current community. If it is greater than the threshold distance (d_{th}) of the current community then the node searches for communities it has recorded previously at a location which is within d_{th} distance from the centroid. All such communities are merged

with current community and removed from the detected communities list. Familiar sets and lists of locations are also merged.

The current community is also checked for the possibility of merging with other communities recorded previously based on *merging criteria* of SIMPLE. If *merging criteria* is satisfied with an existing community, then the current community is merged with the existing community. Familiar sets and lists of locations are also merged.

If merging is not possible, the current community is added as a community detected at the centroid of the locations of current community in the detected community list.

Deciding Current Community

The node then searches for the nearest community recorded from the *current location*. If the distance between its centroid and the *current location* is less than its threshold distance, it is considered as the current community. Otherwise, the node considers the current community as empty.

It is evident from the DOCD mechanism that it detects overlapping community structure without any additional communication overhead as compared to SIMPLE algorithm. The only assumption that the mechanism makes is that each node can track its own location.

3.2.4 Complexity Analysis

Upon each contact of a node with another node, the communication cost involves transmitting familiar set and local community of the node to the neighbour node. So, the communication cost is $\mathcal{O}(n)$ where n denotes the total number of nodes in the network. This communication cost is that of SIMPLE algorithm. DOCD itself does not have any additional communication cost involved.

Computational Complexity

Upon each contact of a node with another node, following computational cost is involved at the node.

SIMPLE algorithm's *admission criteria* and *merging criteria* involves comparing two lists to count the number of common nodes. So, the computational cost for the same is

$\mathcal{O}(n\log(n))$.

Recording visited locations, calculating centroid of a community, and calculating adaptive threshold distance each involves the computational cost of $\mathcal{O}(m)$ where m is the number of locations visited by a node in a community.

Recording detected community has to parts: i) comparing centroids of already recorded communities with the detected community's centroid which involves the computational cost of $\mathcal{O}(p)$ where p is the number of communities in the network, and ii) checking the possibility of merging with one of the already recorded communities based on *merging criteria* which involves the computational cost of $\mathcal{O}(n\log(n))$.

Deciding current community also involves the computational cost of $\mathcal{O}(p)$.

3.2.5 Performance Analysis

To analyze the performance of DOCD mechanism, we use CAHM mobility model (chapter 2) to generate overlapping community structure and to move nodes as per community structure in ONE simulator. We measure the similarity between input communities generated by CAHM and communities detected by each node using DOCD mechanism. For similarity measurement, we use Jaccard index[61], as done in [2] to evaluate the performance of SIMPLE algorithm. Let Γ_i and Γ_j be sets of nodes in communities C_i and C_j respectively. Let $|\Gamma|$ be the cardinality of the set Γ. Then, Jaccard index (σ_J^{ij}) between C_i and C_j is defined as follows.

$$\sigma_J^{ij} = \frac{|\Gamma_i \bigcap \Gamma_j|}{|\Gamma_i \bigcup \Gamma_j|} \tag{3.1}$$

For a node, let input communities in which it is member be $C_i,\ i = 1, 2, ..., M$ and communities detected by the node using DOCD mechanism be $D_j,\ j = 1, 2, ..., N$. For each input community C_i, we calculate Jaccard index with each of the detected communities D_j; i.e., total $M * N$ Jaccard index values are calculated. Then, these values are sorted in descending order. We propose two different similarity index calculation methods based on this ordered list to measure the overall similarity between the input set of communities and the sets of communities detected by all the nodes in the network which are part of at least two communities.

Jaccard-based Similarity Index (JSI)

We consider input community C_i and detected community D_j with the highest Jaccard index value σ_J^{ij} in the ordered list of Jaccard index values to be matching communities. The corresponding Jaccard index value σ_J^{ij} is removed from the ordered list and this value is considered as similarity value between input community C_i and detected community D_j. We also remove Jaccard index values from the ordered list in which either input community C_i or detected community D_j is involved. This procedure is repeated till the ordered list is empty. If $M = N$, then each of the input and detected communities will match with corresponding detected and input community respectively. If $M > N$, then $M - N$ input communities will remain unmatched. If $M < N$, then $N - M$ detected communities will remain unmatched. Let the size of input community C_i be $S_{C_i}^{com}$, summation of sizes of matching input communities be $S_{C_M}^{com}$, summation of sizes of unmatched input communities be $S_{C_U}^{com}$, summation of sizes of unmatched detected communities be $S_{D_U}^{com}$ and set of matching communities be $\Gamma_M = \{(i,j) : C_i \text{ and } D_j \text{ are matching communities}\}$. Then, average Jaccard index for the node is given by

$$\sigma_J^{avg} = \frac{\sum_{(i,j)\in\Gamma_M}(\sigma_J^{ij} * S_{C_i}^{com})}{S_{C_M}^{com} + S_{C_U}^{com} + S_{D_U}^{com}} \tag{3.2}$$

As it is more important to correctly detect larger communities as compared to smaller communities for efficient communication, Jaccard index is multiplied by the size of the matching input community in the numerator of the equation. To normalize the average Jaccard index between [0,1], in the denominator, summation of sizes of input matching communities, input non-matching communities, and detected non-matching communities is taken where, for non-matching input and detected communities, Jaccard index value is considered as 0.

Further, Jaccard-based Similarity Index (JSI) is mean of σ_J^{avg} of all nodes in the network which are part of at least two communities.

Threshold Jaccard-based Similarity Index (TJSI)

In this method, we consider input community C_i and detected community D_j with the highest Jaccard index value σ_J^{ij} in the ordered list of Jaccard index values to be matching communities if and only if $\sigma_J^{ij} > J_{th}$ where J_{th} is Jaccard index threshold; i.e., we consider

two communities to be matching if and only if their Jaccard index is greater than some threshold. The actual Jaccard index between them is not taken into account and is not included in average Jaccard index calculation. We remove the corresponding Jaccard index value σ_J^{ij} from the ordered list. We also remove Jaccard index values from the ordered list in which either input community C_i or detected community D_j is involved. This procedure is repeated till the ordered list is empty. Let size of input community C_i be $S_{C_i}^{com}$, summation of sizes of input communities be S_C^{com}, summation of sizes of unmatched input communities be $S_{C_U}^{com}$, summation of sizes of unmatched detected communities be $S_{D_U}^{com}$ and set of matching communities be $\Gamma_M = \{(i,j) : C_i \text{ and } D_j \text{ are matching communities}\}$. Then, threshold-based average Jaccard index for the node is given by

$$\sigma_J^{T_{avg}} = \frac{\sum_{(i,j)\in\Gamma_M} S_{C_i}^{com}}{S_C^{com} + S_{C_U}^{com} + S_{D_U}^{com}} \tag{3.3}$$

Further, Threshold Jaccard-based Similarity Index (TJSI) is mean of $\sigma_J^{T_{avg}}$ of all nodes in the network which are part of at least two communities.

3.2.6 Simulation Results

There are 200 nodes with the world size of 11000 meters x 11000 meters, divided into a grid of cells of 40 meters x 40 meters each. The transmission range of each node is 40 meters. With a random seed, CAHM generated 13 communities. We run the simulation for 86400 seconds. We have implemented DOCD in ONE simulator.

To see the effect of adaptive threshold distance on the performance, in non-adaptive version of the DOCD mechanism, we do not update initial threshold distance (d_{th}) using the adaptive threshold distance calculation method; i.e. threshold distance for all detected communities remains the same and it is equal to initial threshold distance which is an input parameter to the mechanism. We call the non-adaptive version of the protocol as 'DOCD-fixed'. In Fig. 3.1 and Fig. 3.2, we compare DOCD and DOCD-fixed protocols. From these figures, it is evident that the performance of the mechanism in terms of both Threshold Jaccard-based Similarity Index (TJSI) and Jaccard-based Similarity Index (JSI) is similar. So, in the following analysis of the performance of the mechanism, we consider Jaccard-based Similarity Index (JSI) only. For these results, familiarity threshold value is set to 60 seconds.

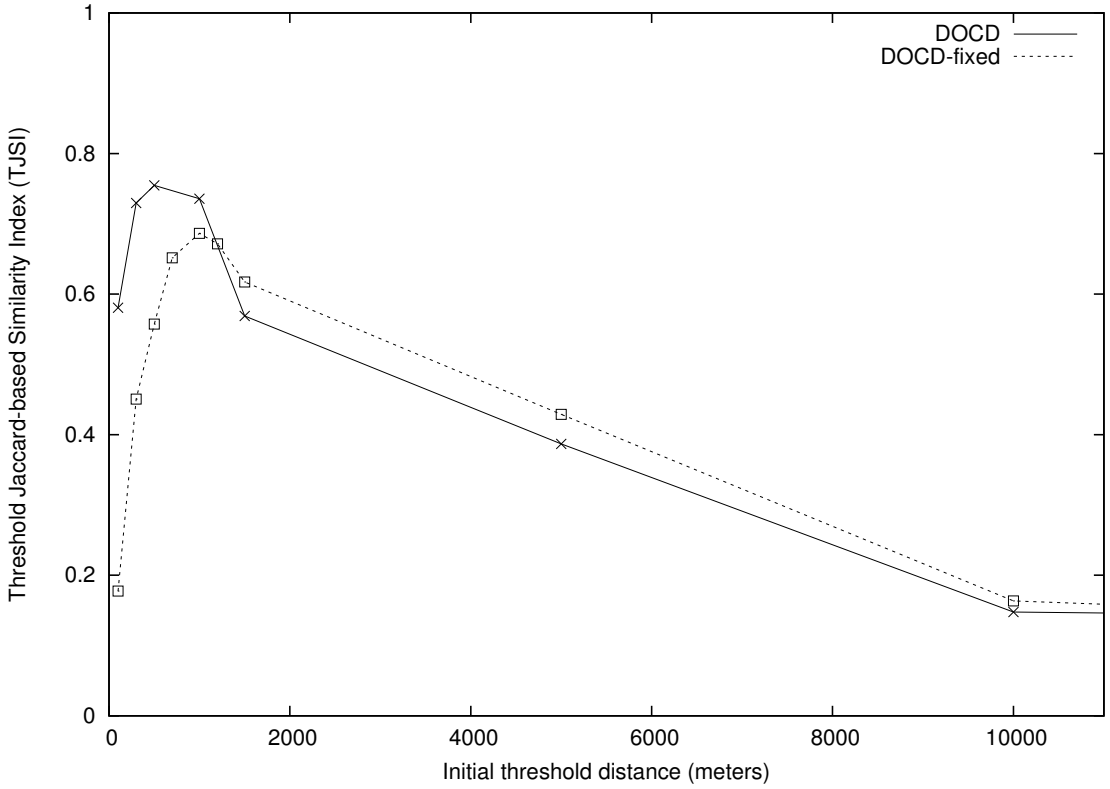

Figure 3.1: Threshold Jaccard-based Similarity Index (TJSI) vs. Initial threshold distance (d_{th})

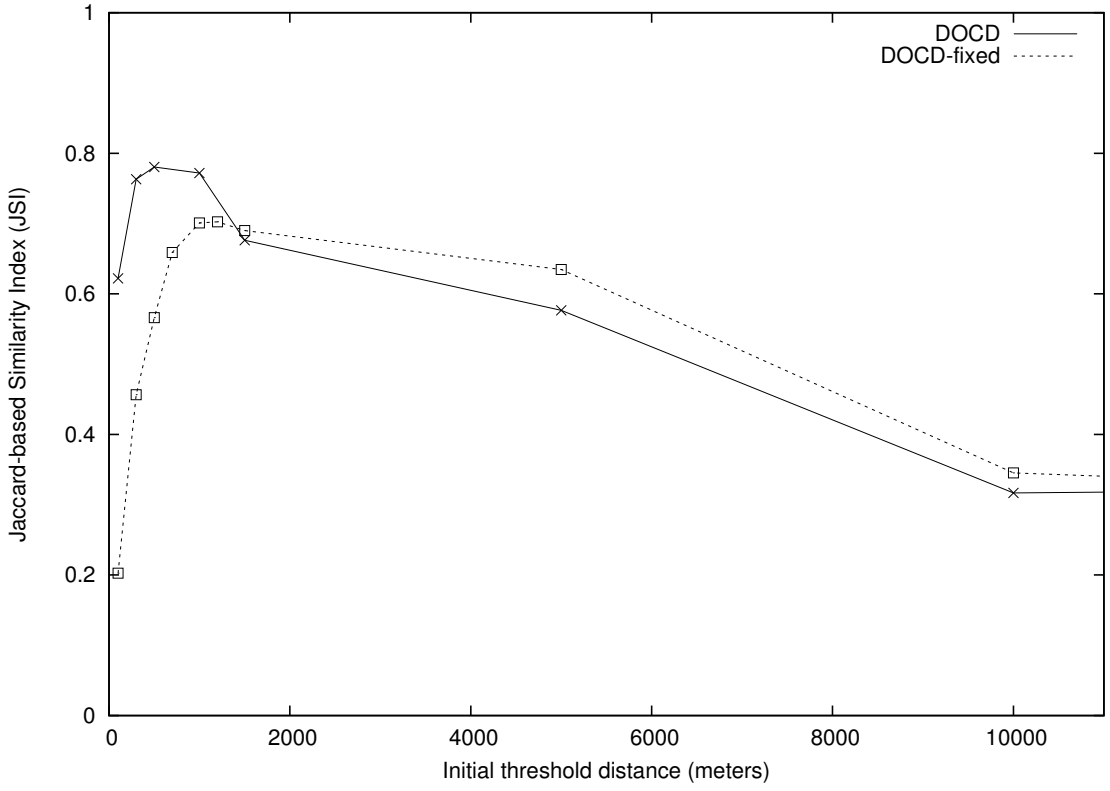

Figure 3.2: Jaccard-based Similarity Index (JSI) vs. Initial threshold distance (d_{th})

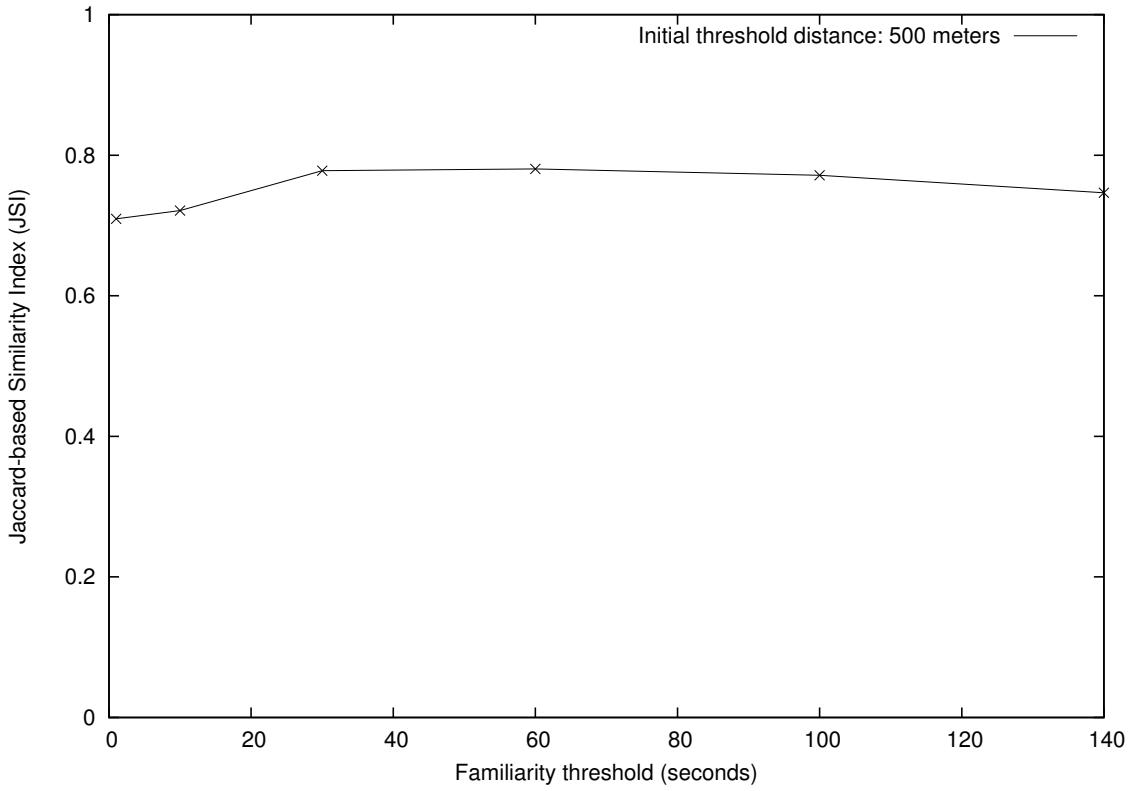

Figure 3.3: Jaccard-based Similarity Index (JSI) vs. Familiarity threshold

As seen in Fig. 3.2, in DOCD-fixed, with very low initial threshold distance (d_{th}), JSI is quite low. While in DOCD, even with very low d_{th}, JSI is acceptable. With low d_{th}, the mechanism breaks single community into multiple communities. Some of these communities get merged due to *merging criteria* of SIMPLE algorithm. Because of this, DOCD increases the threshold distance of the community. As the mechanism also merges all communities which are less than the threshold distance away from the current community, it is able to merge broken multiple communities of an original community because of the increased threshold distance in DOCD. Because of the merging, threshold distance further increases and adapts to the radius of the community in DOCD. But, in DOCD-fixed, merging based on distance does not happen because of non-adaptive threshold distance.

With an increase in initial threshold distance (d_{th}), the performance of the mechanism increases up to some d_{th}. For the given scenario, it is 500 meters for DOCD. Then, it starts decreasing with higher d_{th} value in both the versions. With higher d_{th}, the mechanism merges multiple communities into one. As there is no provision in the mechanism to split merged communities, threshold distance does not adapt with higher d_{th}. So, we rec-

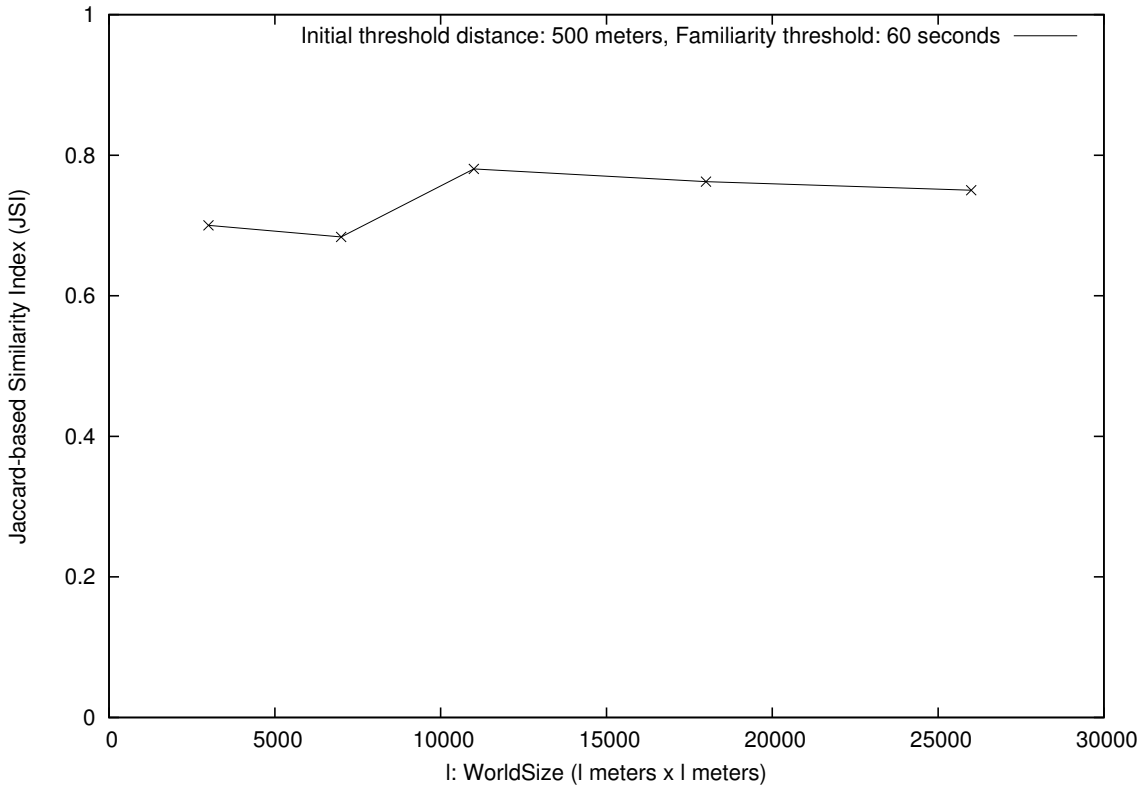

Figure 3.4: Jaccard-based Similarity Index (JSI) vs. World size

ommend keeping initial threshold distance (d_{th}) on the lower side so that the mechanism can adapt it.

Further, as seen in the figure, DOCD achieves maximum similarity index between 75-80%; i.e. it is able to detect overlapping community structure quite accurately.

As shown in Fig. 3.3, the performance of DOCD is not sensitive to the familiarity threshold parameter. As discussed before, familiarity threshold is the parameter of SIMPLE algorithm being used in DOCD which decides minimum cumulative contact duration required between two nodes to be in the familiar set of each other. As seen in the figure, familiarity threshold can be set to as low as 30 seconds in DOCD.

As shown in Fig. 3.4, Jaccard-based Similarity Index (JSI) decreases if world size is very small. As in that case, the same number of communities with the same community structure are placed in a small area. So, communities come very close to each other. As world size increases, the performance improves and then after some world size, it remains almost constant.

We calculate JSI for different overlapping community structures generated by CAHM using different random seeds. With random seeds $\{S_i : i = 1, 2, ..., 5\}$, let the set of

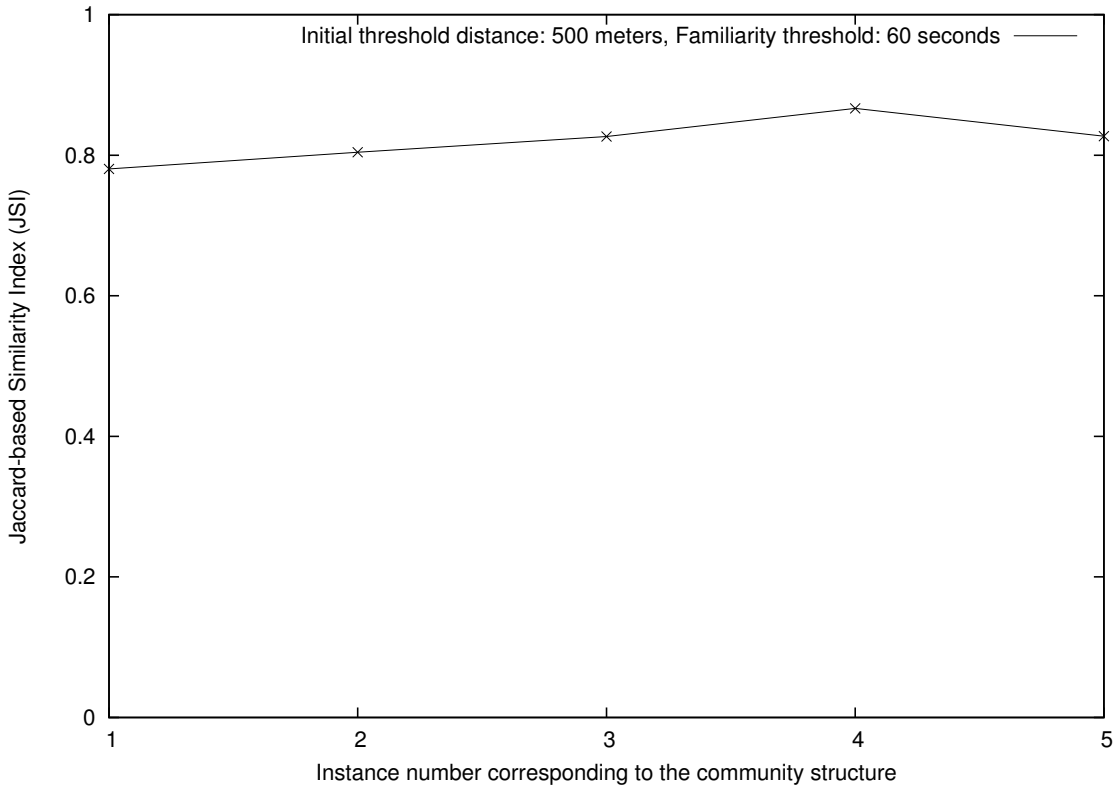

Figure 3.5: Jaccard-based Similarity Index (JSI) vs. Instance number corresponding to the community structure

overlapping community structures generated be $CS = \{CS_i : i = 1, 2, ..., 5\}$. We call i as the instance number corresponding to the community structure CS_i. As shown in Fig. 3.5, the performance of DOCD is similar for different overlapping community structures.

3.3 Analysis of Overlapping Community Structure

Analysis of properties of overlapping community structure formed by human mobility can give important insights for designing better routing protocols. We analyze properties such as actual community sizes, probability of community existence and fraction of hub and gateway nodes present in the community on an average.

We created 172 separate contact graphs based on the number of encounters between all pairs of nodes for 172 intervals of 1000 seconds. An edge between two nodes is added if the number of encounters is greater than some threshold. We imported this graph in Gephi[62]. Then, we ran modularity algorithm, which is an implementation of the algorithm presented in [63] with resolution=1[64], to find out communities. Fig. 3.6

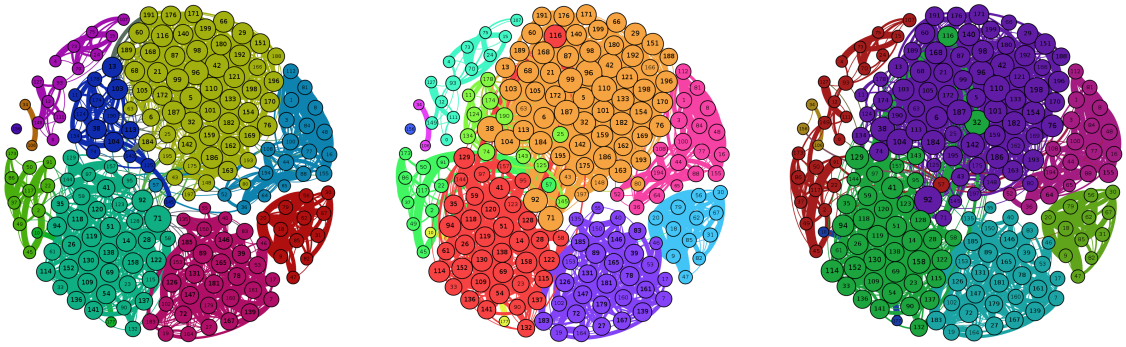

Figure 3.6: Community structure detected by modularity algorithm in three consecutive time intervals using Gephi

shows communities found in first three consecutive time intervals. Nodes with the same colour belong to the same community. The size of a node is proportional to node degree. A node with a different color in a cluster of differently colored nodes means that the node's community has changed.

We checked the similarity between communities generated by CAHM mobility model and communities found using modularity algorithm by calculating Jaccard-based Similarity Index (JSI) (see sub-section 3.2.5). The JSI value came out to be 0.95 which shows that both community sets are highly similar, i.e. CAHM is successful in creating intended community structure of mobile nodes.

Fig. 3.7 shows average of sizes of each community in 172 different intervals. As in CAHM, a node can be part of multiple communities but modularity algorithm will put it with only one of these communities, community sizes found through modularity algorithm are less than community sizes generated in CAHM.

Fig. 3.8 shows that smaller communities are transient and they get merged with bigger communities most of the time. For this, we found community structures using modularity algorithm for 172 different intervals and counted how many times a community is detected by the algorithm. From the figure, it is evident that for 13 communities and 200 total nodes with power-law community sizes, knee point is around community size of 20. The result is very useful for community-based forwarding as it serves as a guideline for deciding whether a group of nodes should be considered as part of a separate community or not.

Fig. 3.9 shows the average of the fraction of identified hub nodes present in communities in different intervals where community structure is created for 1000 nodes. We define a fraction of the nodes of a community which have higher number of neighbour nodes

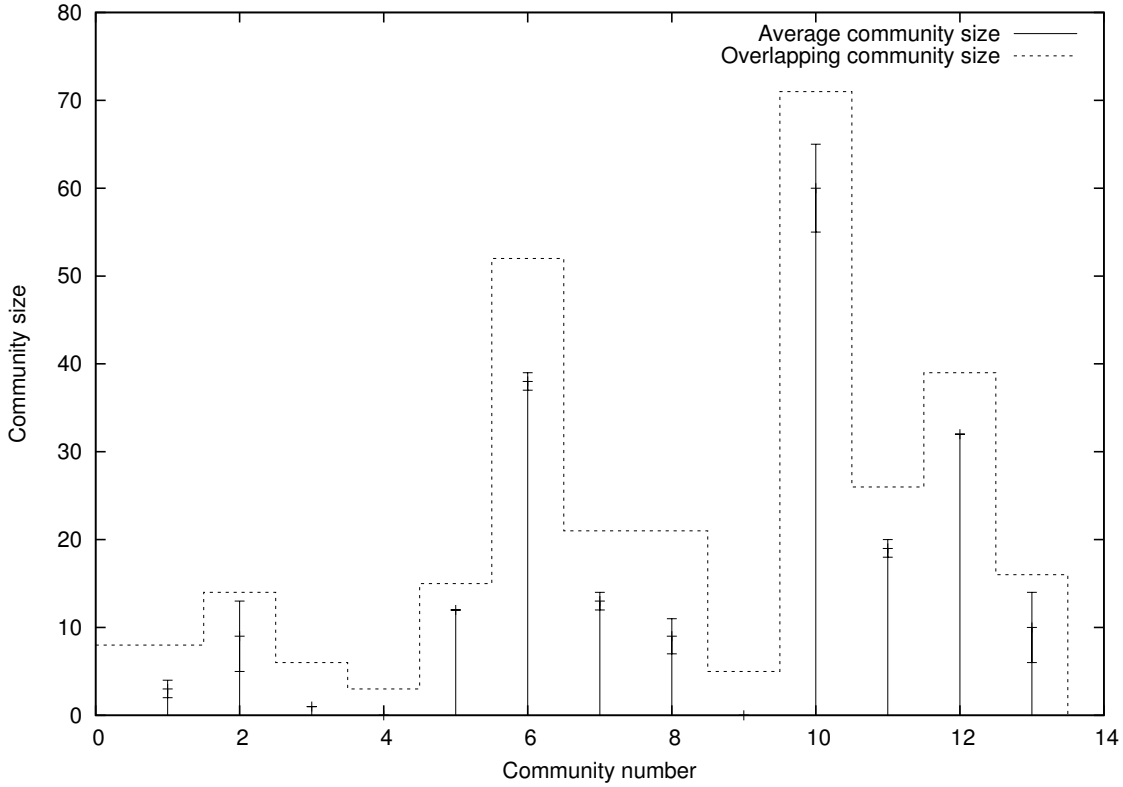

Figure 3.7: Community size vs. Community number

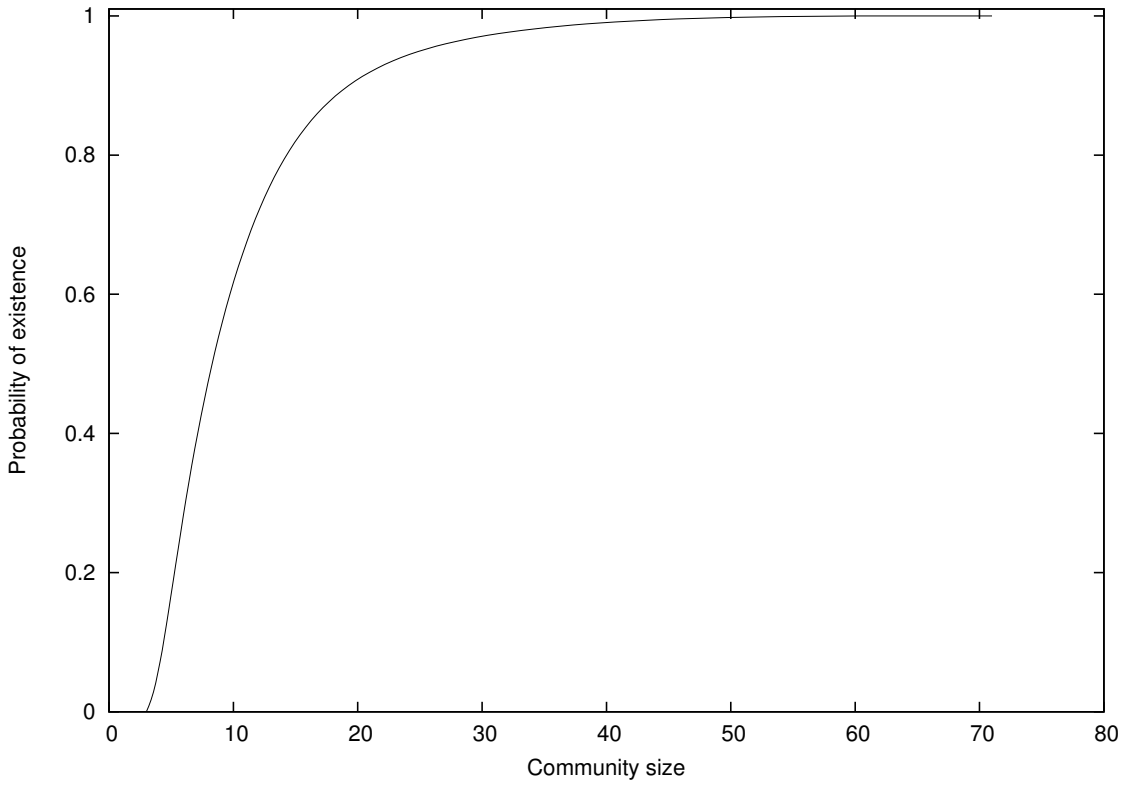

Figure 3.8: Probability of existence vs. Community size

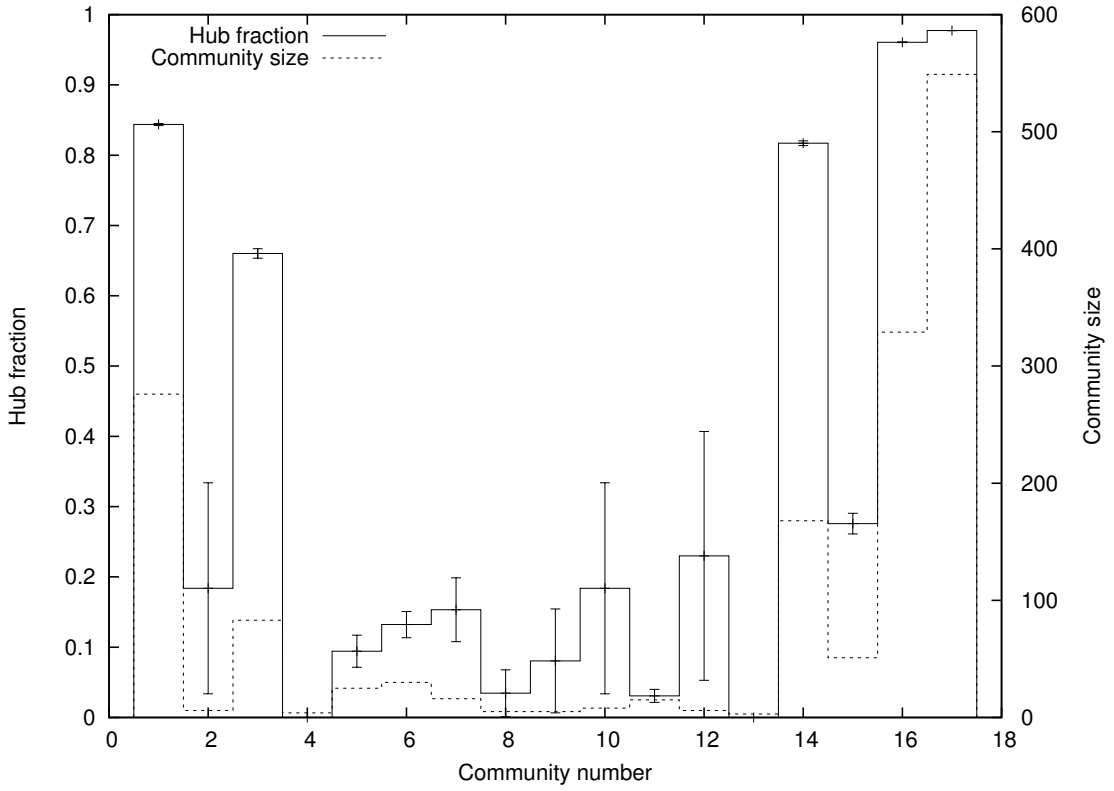

Figure 3.9: Hub fraction vs. Community number

(node degree) in the community as compared to other nodes of the community as hub nodes (locally popular nodes) of the community. To identify neighbour nodes through simulation, we count the number of encounters between all pairs of nodes in a community. Let, for a pair of nodes, the average number of such encounters be μ_e and variance be σ_e. Then, the node's neighbours are the nodes in the community it has encountered more number of times than $\mu_e + \sigma_e$. For this result, we identify first 30% of nodes of a community which have higher number of neighbour nodes in the community as hub nodes. Community size is shown on the second y-axis. It is evident that a higher fraction of hub nodes remain present in larger communities with less standard deviation and a lower fraction of hub nodes remain present in smaller communities with high standard deviation. This information can be used while deciding the percentage of nodes to be considered as hub nodes and also for other forwarding decisions.

Fig. 3.10 shows the average of the fraction of identified gateway nodes present in communities in different intervals. We define a fraction of the nodes of a community which have higher betweenness centrality as compared to other nodes of the community as gateway nodes (globally popular nodes) where betweenness centrality of a node is

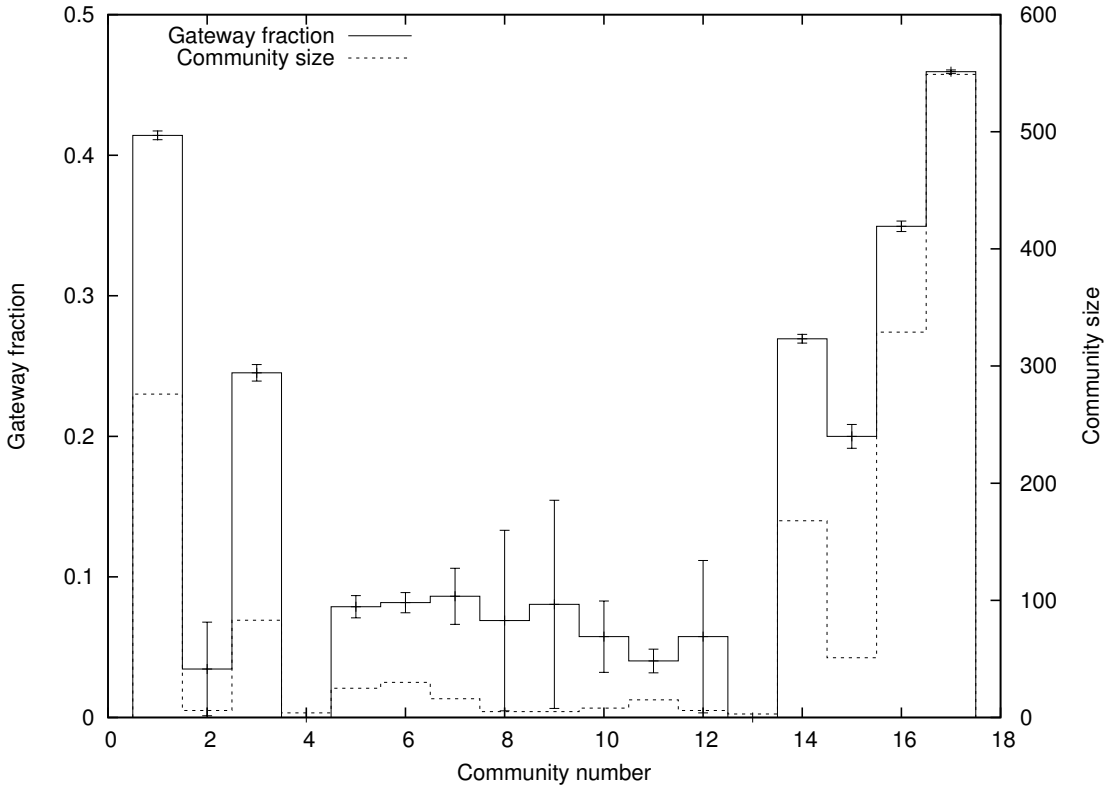

Figure 3.10: Gateway fraction vs. Community number

defined as the number of shortest paths from all nodes of the community to all others nodes in the network that pass through the node. To calculate betweenness centrality, we accumulate encounter information of all nodes through simulation. We form a weighted graph from this encounter information where nodes are vertices in the graph and an edge is placed between two vertices if there is at least one encounter between corresponding nodes. The weight of an edge represents the number of encounters between corresponding nodes. From the resultant weighted graph, for each community, we calculate betweenness centrality values of all nodes with membership in multiple communities. While calculating betweenness centrality for a node, we consider only those shortest paths for which source is in one community and the destination is in another community. For this result, we identify first 30% of nodes of a community which have higher betweenness centrality as gateway nodes. The result is similar to the result for hub nodes. Further, it is evident from Fig. 3.9 and Fig. 3.10 that the fraction of gateway nodes present in a community is much lower than the fraction of hub nodes present in the community.

3.4 Conclusion

Routing protocols in MSN can exploit overlapping community structure formed by humans for efficient forwarding. Further, as discussed in chapter 4, this structure can also be used to identify hub and gateway nodes of a community without doing message flooding. Nodes in MSN need to detect overlapping community structure in a decentralized manner. There is no mechanism available in the literature for the same. We modify existing distributed algorithm SIMPLE and propose Distributed Overlapping Community Detection (DOCD) mechanism to detect overlapping community structure. Simulation results show that DOCD detects overlapping community structure with 75-80% accuracy. Further, it is evident from the results that the performance of DOCD is good if the value of the initial threshold distance (d_{th}) is kept low, and it is not sensitive to the familiarity threshold parameter.

Our analysis of overlapping community structure establishes that small communities are transient. As per simulation results, the threshold for the same is around 10% of total number of nodes in the network. Further, a higher fraction of hub and gateway nodes remain present in larger communities with less standard deviation as compared to smaller communities. Also, the fraction of gateway nodes present in a community is much lower than the fraction of hub nodes present in the community. These results give important insights for designing better forwarding protocols for MSN.

Chapter 4

Identifying Hub and Gateway Nodes in Overlapping Community Structure

As stated previously, different individuals have heterogeneous local and global popularity within a community and in a social network respectively. These locally and globally popular nodes can play very important role in the efficient dissemination of information in Mobile Social Network (MSN)[5]. We call them hub and gateway nodes respectively.

We discuss existing methods to detect hub and gateway nodes in section 4.1. We propose methods, based on mathematical models, to identify hub and gateway nodes from overlapping community structure itself without doing message flooding in section 4.2. In section 4.3, we present the validation of our methods to identify hub and gateway nodes. We conclude the chapter in 4.4.

4.1 Survey of Hub and Gateway Nodes Identification in Community Structure

In BUBBLE Rap[5], nodes with high betweenness centrality in a community are considered as hub nodes where betweenness centrality of a node within a community is equal to the number of shortest paths from all nodes to all other nodes of the community that pass through that node. To calculate betweenness centrality, messages are flooded between

nodes of the community. A node's betweenness centrality in a community is equal to the number of times the node is part of the shortest paths taken by messages sent between nodes of the community. Similarly, to find gateway nodes, messages are flooded between nodes of different communities. But, this method involves considerable overhead. To reduce overhead, the paper also proposes to use the cumulative average of unit-time node degree based on past encounters. The method is called C-Window. But, as shown in the paper, this approximation reduces protocol performance. Wehmuth and Ziviani in [26] propose a method to calculate closeness centrality of nodes in the network by restricted flooding of messages up to given number of hops where closeness centrality of a node within a community is defined as the inverse of the sum of distances to all other nodes in the community. They show that the correlation between optimal ordering of nodes based on centrality values and the ordering obtained using 2-hop flooding is comparable.

Yoneki et al. in [27] consider nodes with high closeness centrality as hub nodes. Gateway nodes are not considered. To calculate closeness centrality, nodes accumulate encounter information. They also exchange this information with community members. Then, each node calculates its closeness centrality from the contact graph thus acquired. Similarly, LocalCom[28] also uses such contact graph and initially considers all nodes which have neighbours in two or more communities as gateway nodes. To prioritize them, it also calculates betweenness centrality of a node from the contact graphs of communities in which the node has neighbours.

All the methods discussed above require either flooding of messages or forwarding of not only node's encounter information but also accumulated encounter information from other nodes. So, most of these methods do not scale with network size or community size. We identify hub and gateway nodes from the overlapping community structure itself without doing message flooding or forwarding of accumulated encounter information from other nodes. The overlapping community structure is found in a distributed manner using Distributed Overlapping Community Detection (DOCD) mechanism proposed in chapter 3. Social-aware routing protocols such as BUBBLE Rap[5] need to find such structure for efficient forwarding anyway. We make use of this structure for identifying hub and gateway nodes also. Further, in DOCD also, a node sends only its own encounter information to its neighbour and does not send accumulated encounter information from other nodes.

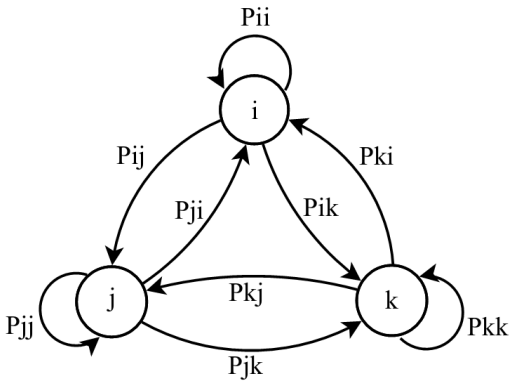

Figure 4.1: Communities and movement of a node between communities represented as Markov chain

4.2 The Proposed Markov Chain-based Methods

Each node of a community estimates its local and global popularity values for the community independently using following information: i) community sizes of all communities in which the node is a member and ii) locations the node visits in these communities. This information is provided by DOCD.

Using this information, we model node's movement in the overlapping community structure as a Markov chain. Communities are considered as states in Markov chain. Fig. 4.1 shows Markov chain of a node with membership in communities i, j, and k. In the figure, P_{xy} with $x = i, j, k$ and $y = i, j, k$ represent probability with which the node travels from community x to community y or remains in the same community ($x = y$) in the next flight. If we can find these probabilities, then steady state probability vector of the node will give the fraction of time for which a node will be in each community. If locations of the communities are known then we can calculate distances between them. It is known that human flight distances follow power-law distribution[14]. We can use this property to find transition probabilities.

Let a node be a member of M communities and let it be currently in community $i = 0$. Let other communities be numbered from 1 to $M - 1$ in the increasing order of their distances from community 0 where the distance between community i and j (d_{ij}) is defined as the distance between centroids of communities i and j. For a node, the centroid of a community is the geometric center of all the locations of the community the node visits. The Distributed Overlapping Community Detection (DOCD) mechanism,

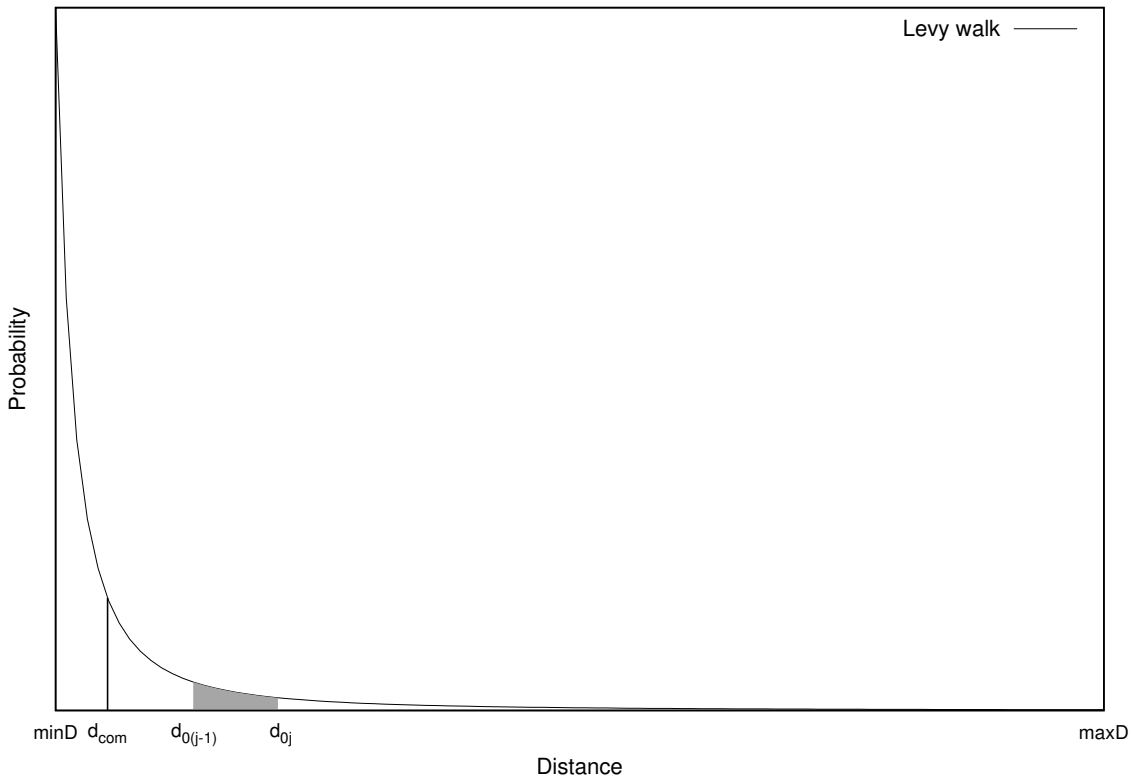

Figure 4.2: Probability Distribution Function (PDF) of distances a node travels

proposed in chapter 3, records locations a node visits in a community while detecting communities of the node and it also computes centroids of the node's communities.

Let probability of node movement from community i to community j be P_{ij}, minimum and maximum distances a node travels in a flight be $minD$ and $maxD$ respectively, d_{com} be the maximum distance the node travels to reach a location within its community and D be power-law exponent. Then, from Fig. 4.2, it is clear that

$$P_{00} = \int_{minD}^{d_{com}} cx^{-D}dx$$
$$P_{0j} = \int_{d_{0(j-1)}}^{d_{0j}} cx^{-D}dx; j = 1, 2, ..., M-1 \quad (4.1)$$
$$\text{Where } c = \frac{1}{\int_{minD}^{maxD} x^{-D}dx} \text{ is the normalizing constant}$$

Here, we assume that all distances between community 0 and locations of each community are either less or greater than all distances between community 0 and locations of the rest of the communities. It is a reasonable assumption to make as distances between different places a person visits daily from home (for example) are rarely same; i.e. dis-

tances between home and locations she visits in her office are either less or greater than distances between home and locations she visits in a park.

Let $\mathbf{w}$ be the steady state probability vector of this Markov chain and the transition probability matrix of the node be $\mathbf{P}$, then it is known that[65]

$$\begin{aligned} w_0 + w_1 + ... + w_{M-1} &= 1 \\ \mathbf{wP} &= \mathbf{w} \end{aligned} \tag{4.2}$$

From these equations, steady state probability vector $\mathbf{w}$ can be found which represents the fraction of time for which the node will be in each community. We use transition probabilities and steady state probability vector thus found to identify hub and gateway nodes.

4.2.1 Hub Nodes

A node with a high local degree in a community can be considered as a hub node of the community where local degree of a node within a community is defined as the number of neighbours of the node in the community. But in overlapping community structure, if a node with high local degree in a community is also member of other communities, then it is important to consider what fraction of time will it remain in the community; i.e. a node's local popularity in a community is dependent not only on local degree but also on the fraction of time it spends in that community.

We consider the number of places a node visits in community i as its local degree (Δ_i), as a node visiting more number of places in a community will come in contact with more number of nodes in the community. We have already calculated the fraction of time the node spends in community i, i.e. w_i. Then, the local popularity (L_i) of the node in community i is

$$L_i = w_i \times \Delta_i \tag{4.3}$$

4.2.2 Gateway Nodes

Intuitively for a node to qualify as a gateway node of a community, it should move from one community to another community frequently. It should also spend more time in that community to carry packets of the community to other communities. Moreover, a node with membership in larger communities should be preferred over a node with membership in smaller communities. Hence, the global popularity of a node should depend on all these three factors.

Mathematically, a node with less average self-transition probability will move between communities more frequently. Let the average self-transition probability of a node be P_{avg} and M be the number of communities in which the node is a member. Then,

$$P_{avg} = \frac{1}{M} \sum_{i=0}^{i=M-1} P_{ii} \tag{4.4}$$

Similarly, a node with high steady state probability w_i for community i will spend more time in community i. Let $\widetilde{S}_i$ represent the summation of sizes of communities in which a node is a member except community i. Then, the global popularity (G_i) of the node in community i is

$$G_i = (1 - P_{avg}) \times w_i \times \widetilde{S}_i \tag{4.5}$$

Using the above methods, each node estimates its local and global popularity values for all communities in which it is a member.

When two nodes come in contact with each other, they exchange their local and global popularity values for all communities in which both are members. Thus, a node accumulates local and global popularity values of member nodes of a community. From the received local and global popularity values, each node independently prepares sorted lists of locally and globally popular nodes for each community in which it is a member. Then, it identifies a given percentage of nodes from these sorted lists of each community as hub and gateway nodes of the community independently.

4.2.3 Complexity Analysis

Our proposed methods to identify hub and gateway nodes do not involve any additional communication cost for calculating local and global popularities at each node. To identify hub and gateway nodes in a community, member nodes of a community exchange and accumulate local and global popularity values. So, the communication cost is $\mathcal{O}(C^3)$ where C is the maximum community size in the network.

The baseline method of flooding in the entire network for identifying hub and gateway nodes based on betweenness centrality involves communication cost of $\mathcal{O}(n^3)$ where n is the number of nodes in the network. So, our method scales with number of nodes in the network but the baseline method is not scalable.

Computational Complexity

In our methods, each node calculates transition probability matrix $\mathbf{P}$, which involves computational cost of $\mathcal{O}(C^2)$. For calculating steady state vector $\mathbf{w}$, the cost is $\mathcal{O}(C^3)$. Further, costs involved for calculating P_{avg} and sorting popularity values are $\mathcal{O}(C)$ and $\mathcal{O}(Clog(C))$ respectively.

At each node, the baseline method involves calculating number of times a node is on shortest paths, for all nodes in the network. The computational cost for the same is $\mathcal{O}(n^3)$. Then, the node sorts all nodes in the network based on their betweenness centralities which involves the cost of $\mathcal{O}(nlog(n))$. So, again, our method scales with the number of nodes in the network but the baseline method is not scalable.

4.3 Validation of the Proposed Methods

To validate proposed methods, we use CAHM mobility model (chapter 2) to generate overlapping community structure and to move nodes as per community structure in ONE simulator. There are 200 nodes in a simulation plane of 5000 meters x 5000 meters, divided into a grid of 62,500 cells of 20 meters x 20 meters each. The transmission range of each node is 20 meters. With a random seed, CAHM generated 13 communities.

To validate steady state probabilities found using Eq. 4.2, in simulation, we kept a log of the amount of time a node has spent in each community in which it is a member.

Dividing these times by total simulation time gives the fraction of the time a node has spent in each community. It turns out that, these values are very closely matching with those found using Eq. 4.2 with the average error of only 0.05.

4.3.1 Validation of Hub Identification Method

We compare ordered list of hub nodes of a community generated through our Markov chain-based method with lists generated using following two methods.

1. **Encounter-based:** In the simulation, we count the number of encounters between all pairs of nodes in a community. Let, for a pair of nodes, the average number of such encounters be μ_e and variance be σ_e. Then, the node's local neighbours are the nodes in the community it has encountered more number of times than $\mu_e + \sigma_e$. Number of such local neighbours denotes the node's local popularity. We order nodes based on these popularity values.

2. **Flooding-based:** For each community, we send packets from different sources to different destinations within the same community through epidemic routing in the simulation. Then, we count the number of times a node of the community is on shortest paths followed by packets to reach destinations and order nodes based on these counts.

Then, we use Spearman's rank correlation coefficient (ρ)[66] to compare two ordered lists. Spearman coefficient is a nonparametric measure of statistical dependence between two variables. It assesses how well the relationship between two variables can be described using a monotonic function. Spearman coefficient $\rho = 1$ means that one ordered list is a perfect monotone function of other ordered list.

Average Spearman coefficient (ρ_{avg}) of all communities comes out to be 0.9890 for Markov chain-based method and encounter-based method. Similarly, for Markov chain-based method and flooding-based method, ρ_{avg} is 0.9980. The results confirm that Markov chain-based method identifies correct hub nodes.

4.3.2 Validation of Gateway Identification Method

We compare ordered list of gateway nodes of a community generated through our method with lists generated using following two methods.

1. **Encounter-based:** We accumulate encounter information of all nodes through simulation. We form a weighted graph from this encounter information where nodes are vertices in the graph and an edge is placed between two vertices if there is at least one encounter between corresponding nodes. The weight of an edge represents the number of encounters between corresponding nodes. From the resultant weighted graph, for each community, we calculate betweenness centrality values of all nodes with membership in multiple communities. While calculating betweenness centrality for a node, we consider only those shortest paths for which source is in one community and the destination is in another community. We order nodes based on these betweenness centrality values.

2. **Flooding-based:** We randomly select source-destination pairs such that both are in different communities. We generate a large number of such pairs and send packets from these sources to corresponding destinations through epidemic routing in the simulation. By definition, gateway node of a community should be at the edge of the community. So, to qualify as a gateway node of a community, a forwarding node should be on the shortest path followed by a packet and the next node on the shortest path after the forwarding node should be in different community. We count the number of times each forwarding node of a community satisfy the above criteria and order them based on these counts.

Average Spearman coefficient (ρ_{avg}) of all communities comes out to be 0.9770 for Markov chain-based method and encounter-based method. Similarly, for Markov chain-based method and flooding-based method, ρ_{avg} is 0.9880. The results confirm that Markov chain-based method identifies correct gateway nodes.

4.4 Conclusion

Hub and gateway nodes of a community can play very important role in the efficient dissemination of information in MSN. Existing methods to detect hub and gateway nodes

Table 4.1: Average Spearman coefficient (ρ_{avg})

	Markov chain-based vs. Encounter-based	Markov chain-based vs. Flooding-based
Hub	0.9890	0.9980
Gateway	0.9770	0.9880

require either flooding of messages or forwarding of not only node's encounter information but also accumulated encounter information from other nodes. So, the performance of these methods does not scale with network size or community size. We identify hub and gateway nodes from the overlapping community structure itself without doing message flooding or forwarding of accumulated encounter information from other nodes.

We propose Markov chain-based methods to identify hub and gateway nodes of a community. The methods involve exchange of independently calculated popularity values of a node with community members only. So, the communication cost (as well as the computational cost) of the method is $\mathcal{O}(C^3)$ only, where C denotes maximum community size; i.e. the performance of Markov chain-based methods scales with the number of nodes in the network. While, for the baseline method, the communication cost (as well as the computational cost) is $\mathcal{O}(n^3)$, where n is the number of nodes in the network.

To validate Markov chain-based methods, we also propose two simulation-based approaches to identify hub nodes and another two simulation-based approaches to identify gateway nodes. We compare ordered lists of hub and gateway nodes generated by Markov chain-based methods with ordered lists generated by these methods. As shown in Table 4.1, these ordered lists are highly correlated, i.e. Markov chain-based methods correctly identify hub and gateway nodes.

Chapter 5

Viral Spread in Mobile Social Network

Viral spread (Many-to-all broadcast) in Mobile Social Network (MSN) is an important functionality which can be useful for a variety of applications in MSN such as message broadcast, news spread, traffic updates etc. Further, it is also useful for routing, as many routing protocols require flooding of control messages. In this chapter, we propose Community Aware Viral Spread (CAVS) protocol for many-to-all broadcast in MSN.

In the following section 5.1, we survey relevant literature. We present our protocol in section 5.2. Section 5.3 discusses simulation results. We conclude the chapter in section 5.4.

5.1 Literature Survey

Maiti et al. in [33] propose use of a directional antenna for epidemic broadcasting in Delay Tolerant Network (DTN). They show that use of directional antenna improves performance. But, mobile devices used by humans do not have a directional antenna. So, their protocol is not applicable in MSN. Yusuke et al. in [34] propose broadcast protocol based on estimation and preservation of stable links in DTN. Their protocol is applicable only when the network is dense which is generally not the case in MSN. Gong et al. in [35] propose a protocol to prioritize broadcast, based on application requirements in DTN; i.e. their protocol does not improve performance in terms of packet delivery ratio or delay.

Table 5.1: Comparison of existing protocols

Protocol	Type of network	Network coding	Community-based	Hub nodes	Gateway nodes
Directional antenna[33]	DTN	No	No	No	No
Stable links[34]	DTN	No	No	No	No
Prioritized broadcast[35]	DTN	No	No	No	No
OSN based broadcast[36]	MSN	No	Yes	No	Yes
Epidemic spread[37]	MSN	No	No	Yes	No
CAVS (our protocol)	MSN	Yes	Yes	Yes	Yes

Lee et al. in [36] propose a broadcast protocol which exploits online social network links to choose nodes for rebroadcasting. The protocol assumes that these nodes are connected with infrastructure based network to deliver messages between disconnected clusters. Connection to the infrastructure-based network is also used to collect information about online social network connections to make better forwarding decisions. While their approach improves average packet delivery delay and delivery ratio, it requires smartphones to have a cellular data connection. Our objective is to develop a viral spread protocol which also works in situations like natural calamities, disaster recovery, and battlefield, when or where such infrastructure based network is not available. Yoneki et al. in [37] analyze the effect of hub nodes for broadcast in MSN. They consider all popular nodes as hub nodes. They conclude that deactivation of hub nodes brings down the performance significantly. So, their aim is not to improve performance but to analyze the effect of hub nodes on the performance.

These protocols are compared in Table 5.1 based on different parameters. We conclude that existing protocols either work with specific assumptions regarding the type of MSN or do not aim to achieve improvement in terms of average packet delivery delay or delivery ratio in the presence of a limited buffer. So, we propose one such protocol. As nodes have limited buffer and using less buffer is also more energy efficient as shown in [32], we use network coding[38] and probabilistic buffering to pro-actively use less buffer while not compromising on performance significantly.

Fragouli et al. in [39] propose network coding based protocol for all-to-all broadcast in ad hoc network. For network coding, a generation is defined as a group of packets such that packets of the same generation only can be linearly combined to generate encoded packets. Packets from different sources are added in a generation in a distributed manner. To decide the generation in which a new packet should be added, each source node inspects generations of incoming packets. If the number of packets in a generation

(generation size) is less than the threshold, the packet is added in that generation. If no such generations are present then a new generation is created. In typically sparse MSN, packets propagate slowly. As a result, nodes do not receive packets of existing generations quickly enough. So, when the average transmission rate is high, source nodes have to create new generations for their packets as they would not have received information about existing generations in the network. As a result, a large number of generations with small generation sizes are created in the network. It leads to less mixing opportunity and less average delivery ratio as packets of the same generation only can be linearly combined. Widmer et al. in [40] use hashing function for generation management and their technique also has the same problem. Therefore, we propose a community-based distributed generation management technique for MSN to overcome this limitation.

For performance comparison, we modify Epidemic routing[3] implementation in ONE simulator by delivering packets to all nodes instead of only to one destination node and by buffering packets probabilistically instead of buffering all the packets for forwarding.

5.2 Community Aware Viral Spread (CAVS) Protocol

The protocol exploits properties of human mobility like overlapping community structure and heterogeneous popularity of nodes for efficient forwarding. Overlapping community structure and popular nodes are found as described in chapter 3 and 4 respectively.

In the viral spread, each node potentially receives packets from all source nodes. If all packets are buffered for forwarding, a node may require a very large buffer. In order to reduce buffer occupancy, all packets should not be buffered at all nodes. Ideally, the decision to buffer a received packet should be based on the number of available copies of the packet in the network. As this information is not easily available, CAVS buffers each packet with some probability. To offset for the loss in performance due to this, CAVS employs network coding. Network coding is a mechanism in which nodes create encoded packets by combining two or more incoming packets and forward encoded packets instead of forwarding incoming packets as it is[38].

For network coding, packets from multiple sources of a community are grouped into

a generation. A packet belongs to a community in which the source node is present at the time of packet creation. Packets of a same generation only can be combined. With an increase in generation size, mixing opportunity increases as there are less number of generations in the network but coding complexity and decoding delay increases. We propose a community-based distributed generation management technique which decides to which generation a new packet of a source is to be added. The technique limits the number of packets contributed by a source to a generation based on target generation size and the estimated number of source nodes in a community.

CAVS exploits overlapping community structure as well as hub and gateway nodes of a community for efficient packet forwarding. For packet transmission, if a node has multiple neighbours then it gives priority to neighbours which are gateway and hub nodes of the current community of the node. Further, while transmitting to a particular neighbour, packets of generations for which the neighbour is a gateway node or the node itself is a gateway node are transmitted first.

Salient features of our protocol are following.

- We exploit social properties of humans such as overlapping community structure and heterogeneous popularity of nodes for efficient forwarding.
- We propose probabilistic buffering of packets at nodes for efficient buffer usage.
- We use network coding to offset the loss of performance due to probabilistic buffering.
- We propose a community-based distributed generation management technique for network coding.
- We suggest optimal values of buffering probability, generation size and percentage of total nodes to be used as hub and gateway nodes in the protocol.

The following sub-sections describe the protocol in detail.

5.2.1 Network Model

Nodes in the network form overlapping community structure. Each community is associated with a place (office, home, park etc.). The location of the place is chosen randomly.

A node can be part of multiple communities. A node moves between different locations within a place when it is in a community. The number of different locations in a community that a node visits depends on its popularity in the community. A node visits places associated with those communities in which it is a member. A node moves within a place more frequently while between places, it moves less frequently. The nodes in the network find such overlapping community structure using Distributed Overlapping Community Detection (DOCD) mechanism proposed in chapter 3. The network is typically sparse and there is generally no contemporaneous path between source and destination. Due to this, we use opportunistic forwarding where packets are exchanged opportunistically when two nodes come within communication range of each other.

5.2.2 Hub and Gateway Nodes

Locally popular (hub) nodes of a community come in contact with more number of nodes in the community than rest of the nodes of the community. Similarly, globally popular (gateway) nodes of a community visit multiple communities and come in contact with more number of nodes in the entire network than rest of the nodes of the community. We use methods described in chapter 4 to identify hub and gateway nodes from overlapping community structure itself without doing message flooding or forwarding of accumulated encounter information from other nodes. These methods order nodes of a community based on their suitability as hub or gateway node. From the ordered list of candidate hub nodes, we choose a fraction of most suitable nodes as hub nodes of the community. Similarly from the ordered list of candidate gateway nodes, a fraction of most suitable nodes are chosen as gateway nodes of the community.

5.2.3 Network Coding

Network coding is a mechanism in which nodes create encoded packets by combining two or more incoming packets and forward encoded packets instead of forwarding incoming packets as it is[38]. Successful reception of a generation does not depend on receiving specific packets of the generation but on receiving sufficient number of independent packets of the generation; i.e. each transmission of a coded packet contributes the same to the eventual delivery of all packets of the generation.

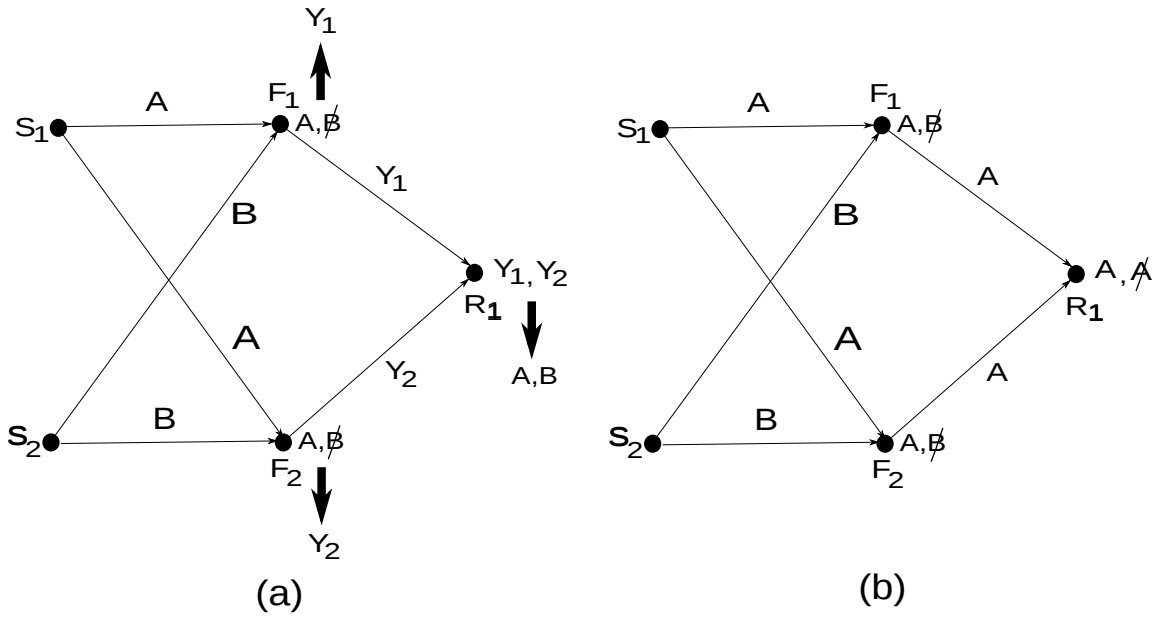

Figure 5.1: Packet forwarding with probabilistic buffering: (a) With network coding (b) Without network coding

Network coding offsets for the performance drop due to probabilistic buffering as shown by an intuitive example in Fig. 5.1. Sources S_1 and S_2 transmit their packets A and B respectively to two nodes F_1 and F_2. In the example, without network coding, nodes F_1 and F_2 receive A and B but both buffer only A for forwarding. Then, both transmit A to R_1. R_1 accepts A from either F_1 or F_2 and rejects A as duplicate from the other. As probability of receiving same packet at R_1 is 0.5, probability of loss at R_1 is 0.5. With network coding, nodes F_1 and F_2 combine A and B before dropping B. Then, F_1 transmits coded packet Y_1 to R_1 and F_2 transmits coded packet Y_2 to R_1. From Y_1 and Y_2, R_1 decodes A and B. So, with network coding, there is no loss at R_1.

In linear network coding, nodes forward linear combinations of two or more input packets as encoded packets. In Random Linear Network Coding (RLNC)[67], coefficients for the linear combination are generated randomly at each node. RLNC is distributed and less computationally intensive than linear network coding. So, we use RLNC in our protocol. The following explanation of RLNC is based on [68].

Random Linear Network Coding (RLNC)

Let each packet contain L bits. If packets to be combined are not of the same size, smaller packets are padded with trailing zeros. Consider s consecutive bits of a packet as

a symbol over the Galois Field $\mathbf{GF}_{2^s}$, i.e. each packet consists of a vector of $\frac{L}{s}$ symbols. An outgoing packet at each node is a linear combination of incoming packets or generated packets at that node where addition and multiplication operations are performed over the Galois Field $\mathbf{GF}_{2^s}$. The encoded packet also contains L bits. An encoded packet contains information about all original packets and multiple such packets can be generated. In effect, information of an original packet is spread across a number of encoded packets.

Encoding Let original packets generated by one or more sources be $M^1, ..., M^n$. Then, an encoded packet $X^1 = \sum_{i=1}^{n} g_i^1 M^i$, where sequence of coefficients $g_1^1, .., g_n^1$ are chosen uniformly at random over the Galois Field $\mathbf{GF}_{2^8}$. The summation is to be done for every symbol in a vector of L/s symbols in a packet, i.e., $X_p^1 = \sum_{i=1}^{n} g_i^1 M_p^i$, where M_p^i and X_p^1 is the p^{th} symbol of M^i and X^1 respectively. Coefficient vector $g^1 = (g_1^1, .., g_n^1)$ is also sent along with encoded packet X^1. The process is illustrated by following example where $n = 2$, $p = 1$, and addition and multiplication operations are over the Galois Field $\mathbf{GF}_{2^8}$.

$$\mathbf{M} = \begin{pmatrix} M^1 \\ M^2 \end{pmatrix} = \begin{pmatrix} 161 \\ 87 \end{pmatrix}, \mathbf{g} = \begin{pmatrix} g_1^1 & g_2^1 \\ g_1^2 & g_2^2 \end{pmatrix} = \begin{pmatrix} 230 & 33 \\ 155 & 85 \end{pmatrix}$$

$$\mathbf{gM} = \begin{pmatrix} 230 & 33 \\ 155 & 85 \end{pmatrix} \begin{pmatrix} 161 \\ 87 \end{pmatrix} = \begin{pmatrix} 108 \\ 104 \end{pmatrix} = \begin{pmatrix} X^1 \\ X^2 \end{pmatrix} = \mathbf{X}$$

Forwarding nodes can also do encoding on already encoded packets. Consider a node has a set $(g^1, X^1), ..., (g^m, X^m)$, where $m \leq n$. The node can generate a new encoded packet (k^1, Y^1) by selecting uniformly at random a set of coefficients $h_1^1, ..., h_m^1$ and computing linear combinations $Y^1 = \sum_{j=1}^{m} h_j^1 X^j$ and $k_i^1 = \sum_{j=1}^{m} h_j^1 g_i^j$ where k_i^1 is the i^{th} element of the coefficient vector $k^1 = (k_1^1, .., k_n^1)$. For following example, $m = 2$.

$$\mathbf{g} = \begin{pmatrix} 230 & 33 \\ 155 & 85 \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 108 \\ 104 \end{pmatrix}$$

$$\mathbf{h} = \begin{pmatrix} h_1^1 & h_2^1 \\ h_1^2 & h_2^2 \end{pmatrix} = \begin{pmatrix} 39 & 202 \\ 236 & 141 \end{pmatrix}$$

$$\mathbf{hX} = \begin{pmatrix} 39 & 202 \\ 236 & 141 \end{pmatrix} \begin{pmatrix} 108 \\ 104 \end{pmatrix} = \begin{pmatrix} 142 \\ 97 \end{pmatrix} = \begin{pmatrix} Y^1 \\ Y^2 \end{pmatrix} = \mathbf{Y}$$

$$\mathbf{hg} = \begin{pmatrix} 39 & 202 \\ 236 & 141 \end{pmatrix} \begin{pmatrix} 230 & 33 \\ 155 & 85 \end{pmatrix} = \begin{pmatrix} 133 & 186 \\ 127 & 123 \end{pmatrix} = \begin{pmatrix} k_1^1 & k_2^1 \\ k_1^2 & k_2^2 \end{pmatrix} = \mathbf{k}$$

Decoding Let a node has received the set $(k^1, Y^1), ..., (k^m, Y^m)$, where $m \leq n$. To retrieve original packets, it needs to solve the system of m linear equations $Y^j = \sum_{i=1}^{n} k_i^j M^i$, where $\{M^i : i = 1, ..., n\}$ is a set of unknowns. If $m = n$, i.e. the number of received packets are equal to the number of original packets, we can recover all the original packets provided all the received packets are linearly independent. But, there is a possibility that some of the encoded packets are linearly dependent. Simulation results indicate that even for small field sizes (e.g. $s = 8$), the probability becomes negligible[69]. For following example, $m = n = 2$.

$$\mathbf{k} = \begin{pmatrix} 133 & 186 \\ 127 & 123 \end{pmatrix}, \mathbf{Y} = \begin{pmatrix} 142 \\ 97 \end{pmatrix}$$

$$\begin{pmatrix} 133 & 186 \\ 127 & 123 \end{pmatrix} \begin{pmatrix} M^1 \\ M^2 \end{pmatrix} = \begin{pmatrix} 142 \\ 97 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} M^1 \\ M^2 \end{pmatrix} = \begin{pmatrix} 161 \\ 87 \end{pmatrix} = \mathbf{M}$$

5.2.4 Community-based Distributed Generation Management

Generations are identified by community Id (CId) and a generation counter (C). Every source node starts with an independent counter value of $C = 0$ for each community in which it is a member. Let the maximum target generation size of the network be G_{max} and number of source nodes in a community be S. The number of source nodes (S) for a community is estimated from source identification of packets of the community present in a node's buffer. As nodes of the same community communicate with each other frequently, S can be predicted quite accurately. Then, threshold (N_{th}) is given by

$$N_{th} = \frac{S}{G_{max}} \tag{5.1}$$

If $N_{th} \geq 1$ then each source node of a community contributes N_{th} number of packets to each generation of the community with same C value before incrementing C by one. Else, each source node of a community contributes a packet to a generation of the community with 'N_{th}' probability.

As, a source node may generate packets with different rate than others and is absent in a community when it visits other communities, counter C of the node for a given

community may be too low as compared to nodes which permanently reside in the community and/or are generating packets with higher rate. If C is too low, the source will add packets to old generations and newly added packets will get little mixing opportunity, as many of the packets of old generations may have already been dropped from buffers of community members. So, to avoid adding packets to old generations, we need to loosely synchronize counters of all nodes of a community such that difference between maximum and minimum values of counters of community members is not greater than some threshold. For the same, maximum counter value ($maxC$) in received generations of the community is recorded. If the difference between counter C of the source node and $maxC$ is greater than some threshold (D_{th}), then C is set to $maxC$.

The entire generation management procedure at a source node for a community is presented in the algorithmic form in Algorithm 5.1. In the algorithm, function $rand()$ generates random number between 0 and 1 with uniform distribution, 'o' is string concatenation operator, GId represents generation Id.

5.2.5 Opportunistic Forwarding

For packet forwarding, a node first decides the number of independent packets it can contribute to its neighbour node, of each generation it has in its forwarding buffer, by exchanging coefficient matrices of generations with its neighbour node. Then, it forwards these packets to its neighbour. If a node has multiple neighbours then it gives priority to neighbours which are gateway and hub nodes of the current community of the node. Further, while transmitting to a particular neighbour, packets of generations for which the neighbour is a gateway node or the node itself is a gateway node are transmitted first. The detailed forwarding protocol is explained in following paragraphs.

A node maintains nodes in its communication range in sorted neighbour list in the following order: all gateway neighbours of the current community, all hub neighbours of the current community, all regular nodes. It also maintains two buffers. One for its own packet reception and decoding and the other for forwarding. We call these buffers 'received' buffer and 'forwarding' buffer respectively. A node also maintains two coefficient matrices for each generation it receives. In one coefficient matrix, it adds coefficients of all independent packets it receives of a generation. We call it 'received' coefficient matrix.

Algorithm 5.1: Generation management at a source node for a community

```
C = 0
N_th = S / G_max
i = 0
for each new packet do
  if maxC − C > D_th then {Synchronize counter C to maximum counter maxC in
  the community}
    C = maxC
    i = 0
    GId = CId o C
    continue
  end if
  if N_th < 1 then {Contribute a packet with 'N_th' probability to a generation}
    while true do
      if rand() < N_th then
        break
      else
        C = C + 1
      end if
    end while
  else {Contribute 'N_th' packets to each generation}
    if i == N_th then
      C = C + 1
      i = 0
    else
      i = i + 1
    end if
  end if
  GId = CId o C
end for
```

In the second coefficient matrix, it adds coefficients of an independent packet only if it decides to add the packet in 'forwarding' buffer. We call it 'forwarding' coefficient matrix.

If a node has packets in its forwarding buffer and if communication medium is not busy, it transmits independent packets to the first neighbour in the sorted neighbour list till no further packet can be transmitted to that neighbour. Then, it tries second neighbour and so on. To explore the possibility of packet transmission to a neighbour, the node sends coefficient matrices of all generations present in its forwarding buffer to the neighbour. From these matrices and based on its own received coefficient matrices, the neighbour derives the number of independent packets of different generations the node can transmit to it and sends this information to the node. The node sends all packets of one generation before switching to next generation. The node decides the order in which generations should be sent as per following forwarding policy.

- The node first sends packets of existing generations at the neighbour. In that, the generation for which minimum number of packets are to be transmitted is selected first.
- Then, it sends packets of generations for which the neighbour is a gateway node; i.e. the neighbour is a gateway node of communities of these generations. In that, the generation which is the oldest in the node's buffer is selected first. We consider creation time of a generation as the creation time of the last packet added in the generation.
- Then, it sends packets of generations for which it is a gateway node and its current community is different from generations' communities. In that, the oldest generation in the node's buffer is selected first.
- Then, it sends packets of the oldest of remaining generations.

As, a packet is sent only if it is useful to a neighbour, only one transmission per packet is done for each receiver in the network.

5.2.6 Packet Reception and Purging

When a node receives a packet, it stores it in its 'received' buffer if it is independent of other received packets of the generation and adds its coefficients in 'received' coefficient

matrix of the generation. Packets of a generation are dropped from 'received' buffer once they are decoded.

The node adds the packet to 'forwarding' buffer with buffering probability (P_b). Otherwise, it just linearly combines the received packet with each of the existing packets of the generation in 'forwarding' buffer, but does not add the packet in 'forwarding' buffer. In network coding, it is beneficial to linearly combine a packet with each of the existing packets of the generation before dropping it.

Coefficient matrices of a generation along with all the packets of the generation in 'received' buffer and 'forwarding' buffer are purged after Time to Live (TTL). For maintaining creation time of a generation and for purging a generation after TTL, we assume that nodes in the network are loosely time synchronized.

If 'forwarding buffer' of a node gets full, a packet of the oldest generation in the buffer is dropped from the buffer. But, before dropping, it is linearly combined with each of the existing packets of the generation.

5.2.7 Complexity Analysis

In our protocol, CAVS, when two nodes come in contact, they exchange coefficient matrices of generations they have in their forwarding buffer. This involves the communication overhead of $\mathcal{O}(G^2)$. Further, a coefficient vector is also included in the header of each data packet for which the communication overhead is $\mathcal{O}(G)$.

In Epidemic routing, when two nodes come in contact, each node sends a data packet identification to the neighbour node. The neighbour node then intimates the sender whether it has already received that data packet or not. If the data packet is not already received then the sender sends it. Otherwise, the sender tries for another packet. So, the communication overhead is $\mathcal{O}(p)$ where p is the number of packets in the node's forwarding buffer.

Computational Complexity

In CAVS, primarily, computational complexity is in encoding and decoding operations of linear network coding. For packet size k, encoding involves the computational cost of $\mathcal{O}(kG)$. For decoding, the cost is $\mathcal{O}(kG^3)$. For Epidemic routing, the computational cost

Table 5.2: Simulation parameters

Number of nodes	200
Simulation area	41000 meters x 41000 meters
Communication range	40 meters
Packet/Bundle size	10 KB
Buffer size	600 KB
Buffering probability (P_b)	0.3
Inter-packet arrival time (T)	220 seconds
Maximum generation size (G)	16
Percentage of nodes as hubs in each community	8%
Percentage of nodes as gateways in each community	20%
Total number of hubs in the network	19
Total number of gateways in the network	24
TTL	400 minutes
Cell size in CAHM	40 meters
Cell multiplier m in CAHM	4

is negligible.

5.3 Simulation Results

We evaluate Community Aware Viral Spread (CAVS) protocol using Community Aware Heterogeneous Human Mobility (CAHM) model proposed in chapter 2 which incorporates properties of human mobility derived from real-world mobility traces as well as from social network theory. We have implemented CAVS protocol in ONE simulator. We compare packet delivery ratio and average packet delivery delay of CAVS and Epidemic routing protocols. We define packet delivery ratio as the ratio of the average number of packets received by each node to the total number of packets sent by all nodes. Similarly, we define average delivery delay as the average of delivery delays of all packets received by all nodes in the network. Simulation parameters are as per Table 5.2 unless otherwise mentioned. The speed of nodes varies based on distance to be traveled from one location to other. The parameters are chosen to represent an application scenario in which all devices of the network in a city transmit sensed environmental conditions of their surrounding places periodically to all other users of the network. All readings are taken once network reached steady state. With a random seed, CAHM generated 14 overlapping communities for 200 nodes.

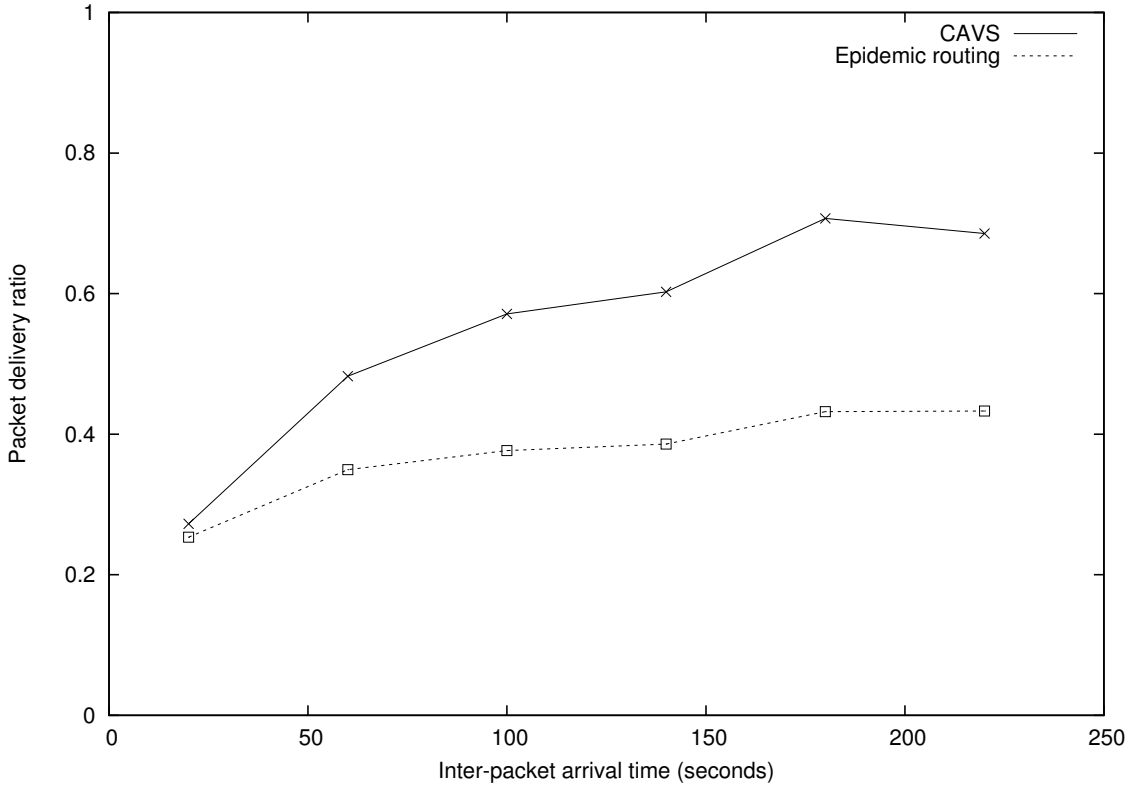

Figure 5.2: Packet delivery ratio vs. Inter-packet arrival time

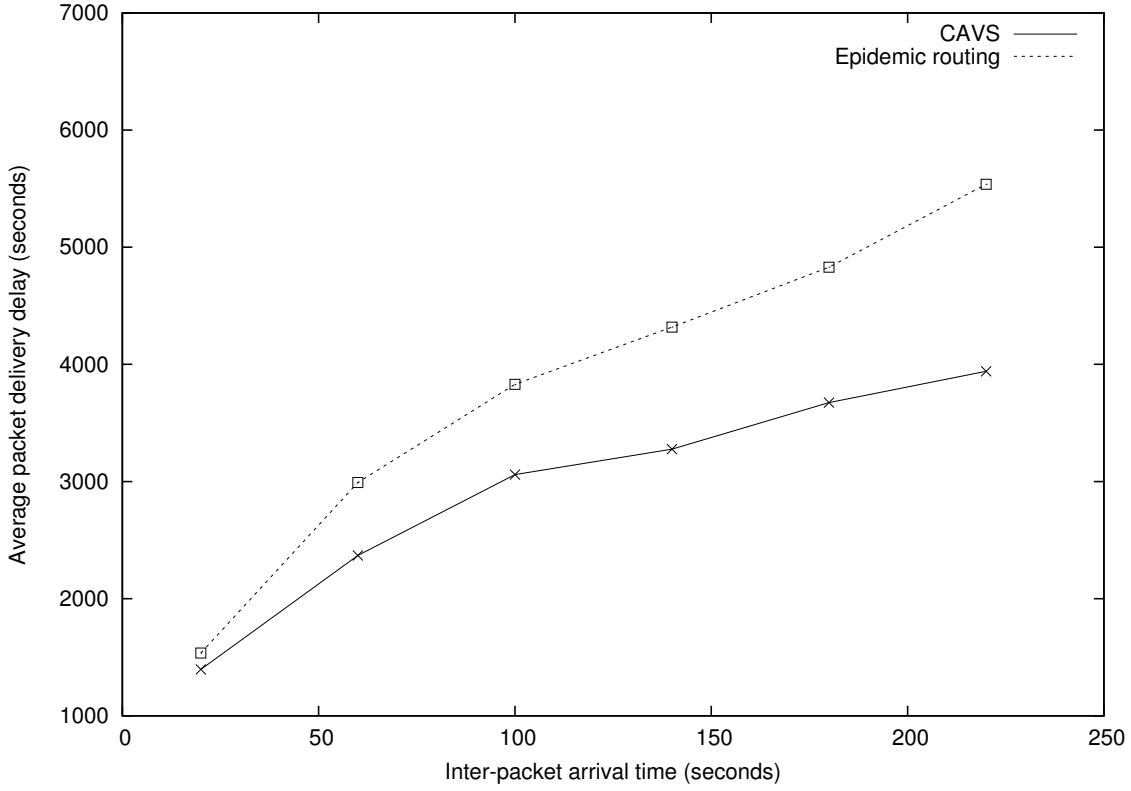

Figure 5.3: Average packet delivery delay vs. Inter-packet arrival time

As shown in Fig. 5.2 and Fig. 5.3, packet delivery ratio and average packet delivery delay of both CAVS protocol and Epidemic routing protocol are similar when the traffic is high. But, as traffic decreases, the performance of CAVS is increasingly better than Epidemic routing. It is because, through the use of network coding as well as hub and gateway nodes, CAVS moves packets faster and reaches more number of remote and less connected nodes as compared to Epidemic routing. But, in high traffic scenario, packets get dropped before they can be delivered to remote and less connected nodes because of buffer overflow (Fig. 5.4). Further, delivering packets to nearby well-connected nodes take similar time in both protocols. So, the performance of CAVS and Epidemic routing is similar when the traffic is high. For inter-packet arrival time of 220 seconds, packet delivery ratio and average packet delivery delay of CAVS is 58% more and 41% less than Epidemic routing respectively. As seen in Fig. 5.3, average packet delivery delay of both protocols increase with a decrease in traffic. It is because, when traffic is high, packet delivery ratio is less as seen in Fig. 5.2; i.e. packets are only delivered to nearby nodes before getting dropped due to buffer overflow and delivering packets to nearby nodes takes considerably less time as compared to remote nodes.

Fig. 5.5 and Fig. 5.6 show the performance of CAVS, CAVS (without hubs and gateways) and Epidemic routing for different buffering probability (P_b) values. For all three protocols, with an increase in P_b, packet delivery ratio increases and average packet delivery delay decreases as expected. As Epidemic routing is optimal without resource constraints, CAVS matches Epidemic routing performance for $P_b = 1$. But, with high P_b, buffer occupancy tends to be high as each node buffers all the packets it receives for forwarding (Fig. 5.7). So, the goal is to operate with as low P_b as possible without compromising on the performance significantly. CAVS gives acceptable performance for P_b value as low as 0.3. With a decrease in P_b, the performance of CAVS gets increasingly better than Epidemic routing. For $P_b = 0.1$, packet delivery ratio and average packet delivery delay of CAVS is 122% more and 22% less than Epidemic routing respectively. Starting from $P_b = 0.1$, the rate of increase in packet delivery ratio and the rate of decrease in average packet delivery delay in CAVS and Epidemic routing protocols are high till $P_b = 0.3$. So, we recommend operating the protocol with $P_b = 0.3$ approximately. The large performance difference between CAVS and CAVS (without hubs and gateways) for lower P_b values shows that hub and gateway nodes play very significant role in the

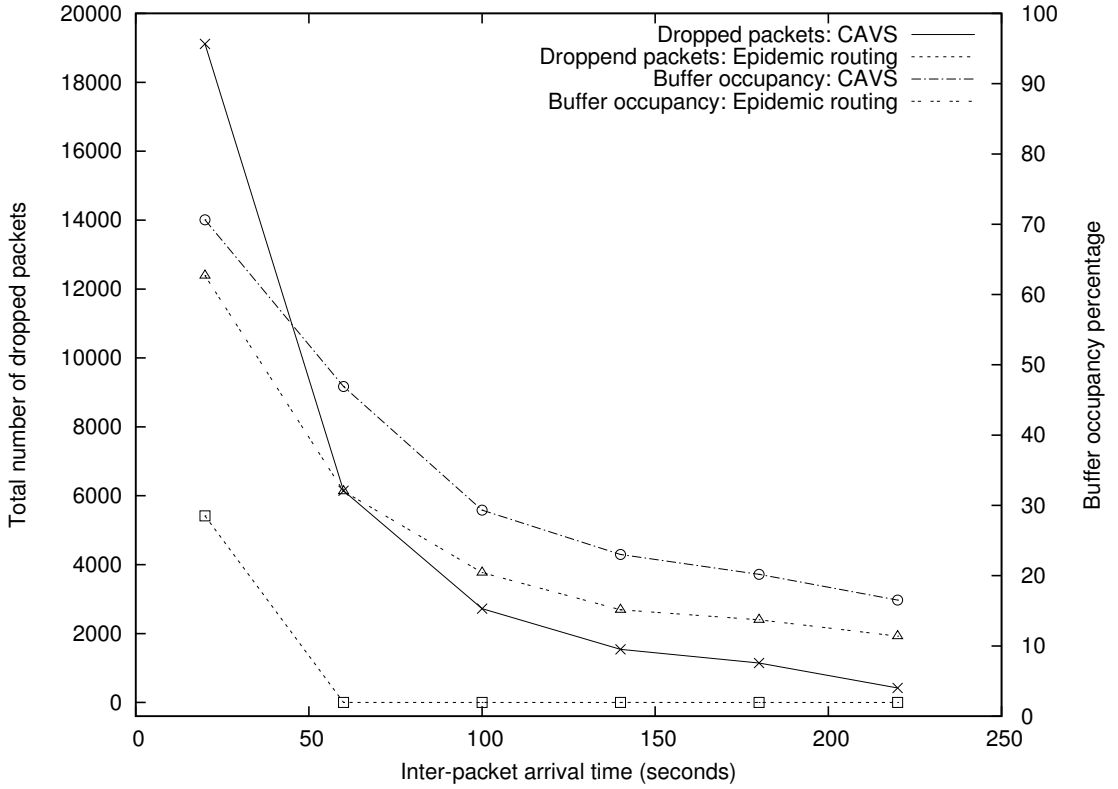

Figure 5.4: Total number of dropped packets and Buffer occupancy percentage vs. Inter-packet arrival time

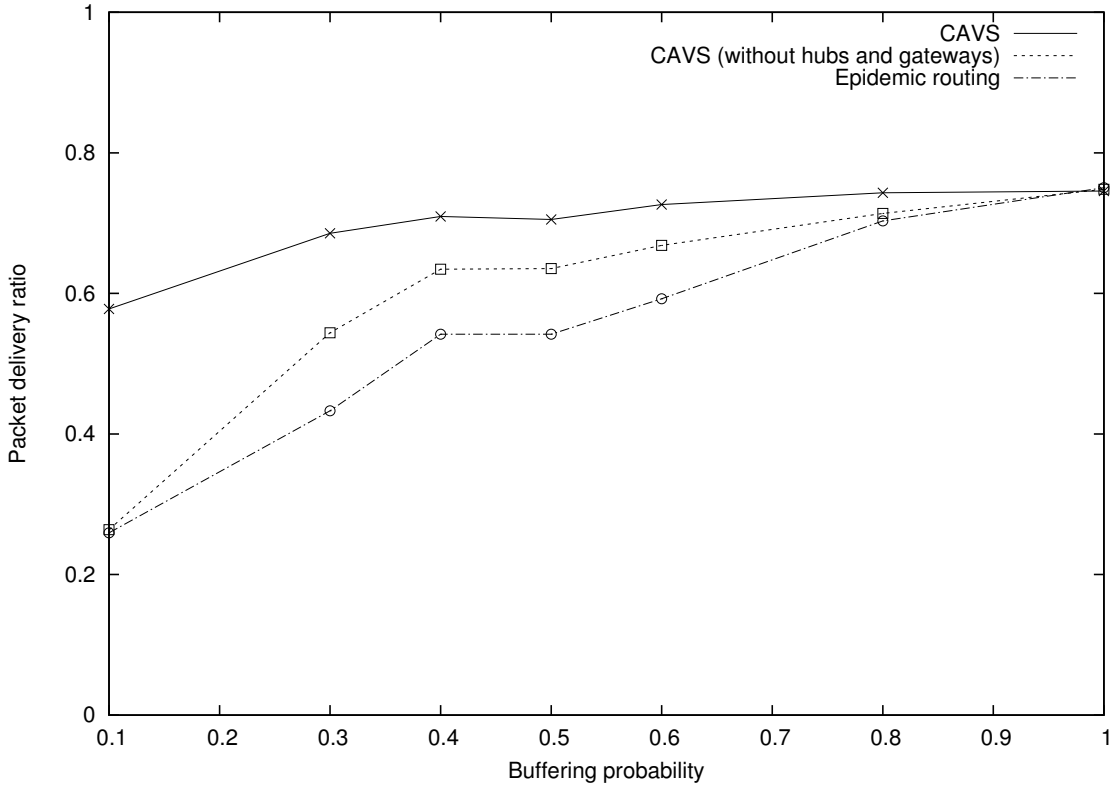

Figure 5.5: Packet delivery ratio vs. Buffering probability (P_b)

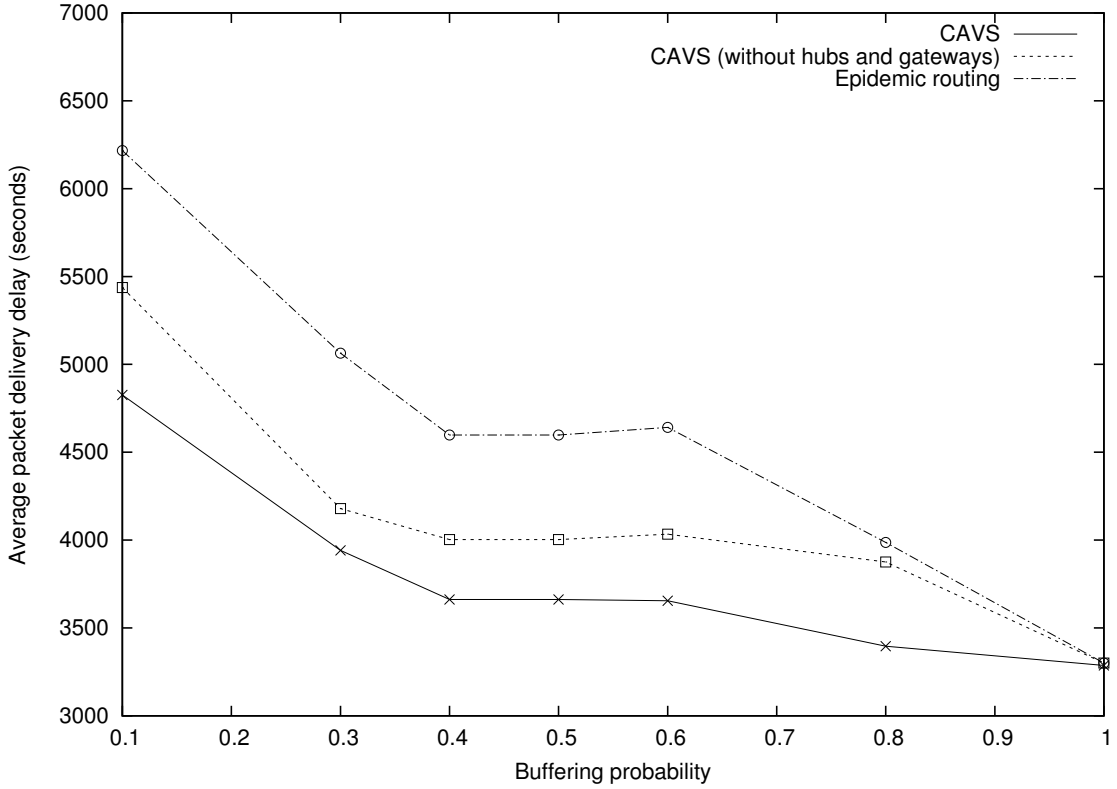

Figure 5.6: Average packet delivery delay vs. Buffering probability (P_b)

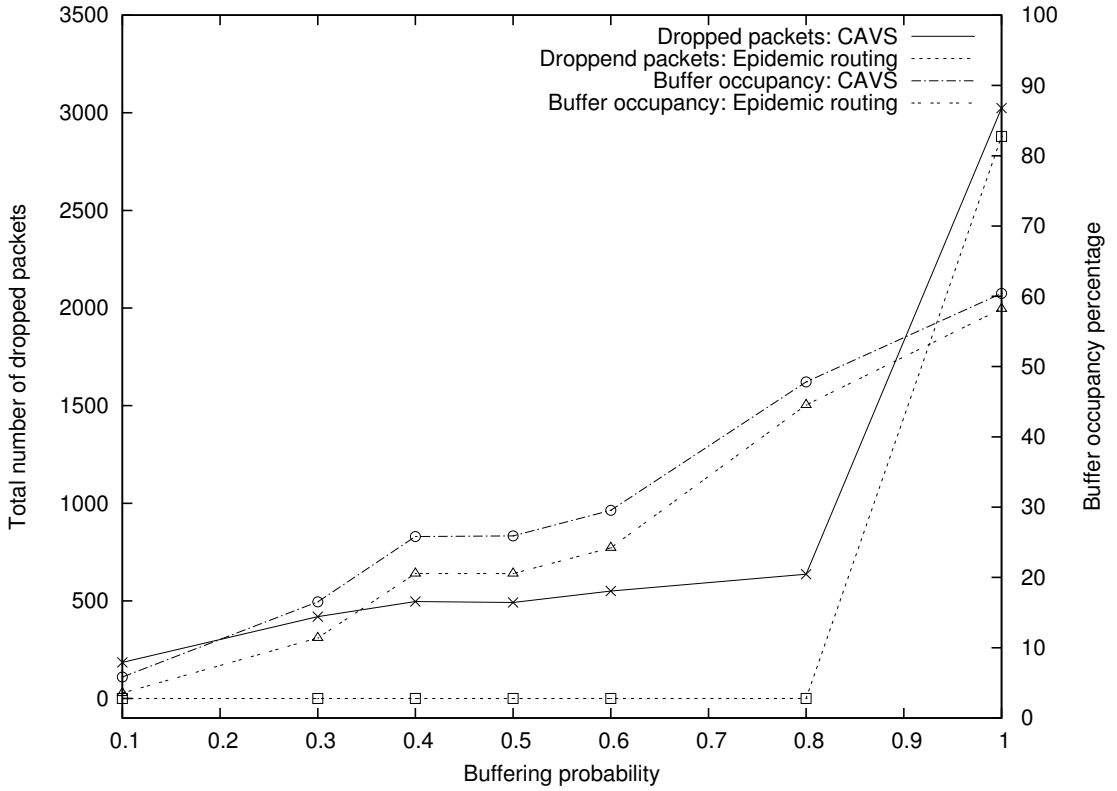

Figure 5.7: Total number of dropped packets and Buffer occupancy percentage vs. Buffering probability (P_b)

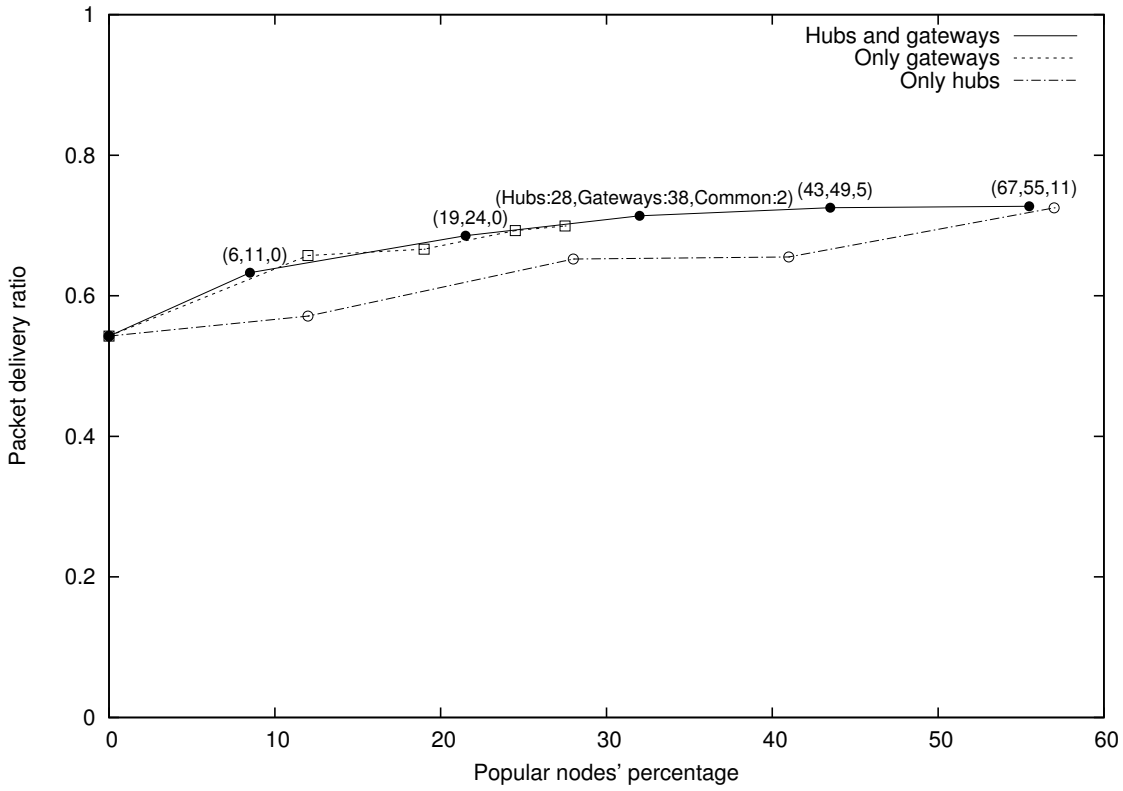

Figure 5.8: Packet delivery ratio vs. Popular nodes' percentage

performance improvement of CAVS protocol.

Fig. 5.8 shows the packet delivery ratio of CAVS with i) both hubs and gateways, ii) only gateways, and iii) only hubs. With only gateways, packet delivery ratio is better than CAVS with only hubs and is similar to CAVS with both hubs and gateways. It shows that gateway nodes play a greater role in the performance improvement of CAVS as compared to hub nodes. But, only the nodes with membership in more than one community can work as gateway nodes and such nodes in the network are limited. In our simulation scenario, these are 27% of total nodes. As seen in the figure, to increase popular nodes' percentage further, CAVS can use more and more hub nodes along with all possible gateway nodes. With an increase in popular nodes' percentage, packet delivery ratio improves. But, the improvement in performance beyond 20% of total nodes as popular nodes is not significant.

As shown in Fig. 5.9, packet delivery ratio is almost independent of the maximum generation size (G). But, as per Fig. 5.10, there is a 12% improvement in average packet delivery delay with $G = 16$ as compared to average packet delivery delay with $G = 2$. Also, after $G = 16$, it remains almost constant. So, we recommend setting maximum

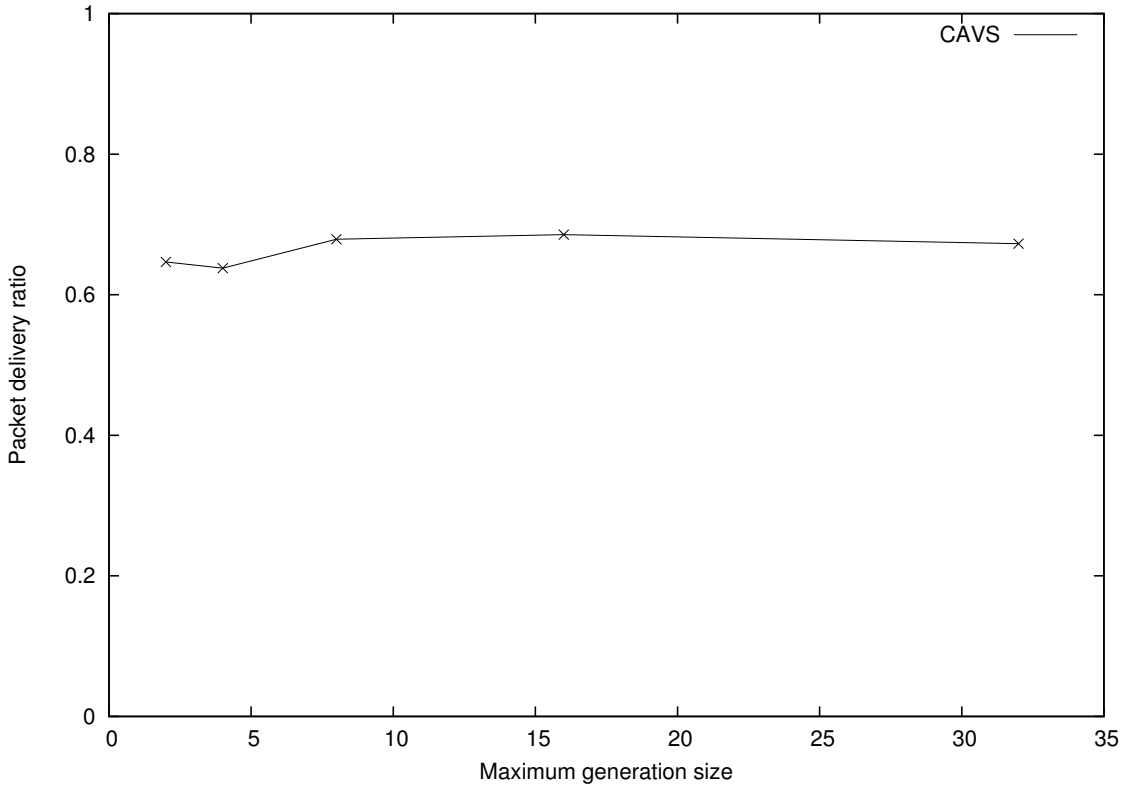

Figure 5.9: Packet delivery ratio vs. Maximum generation size (G)

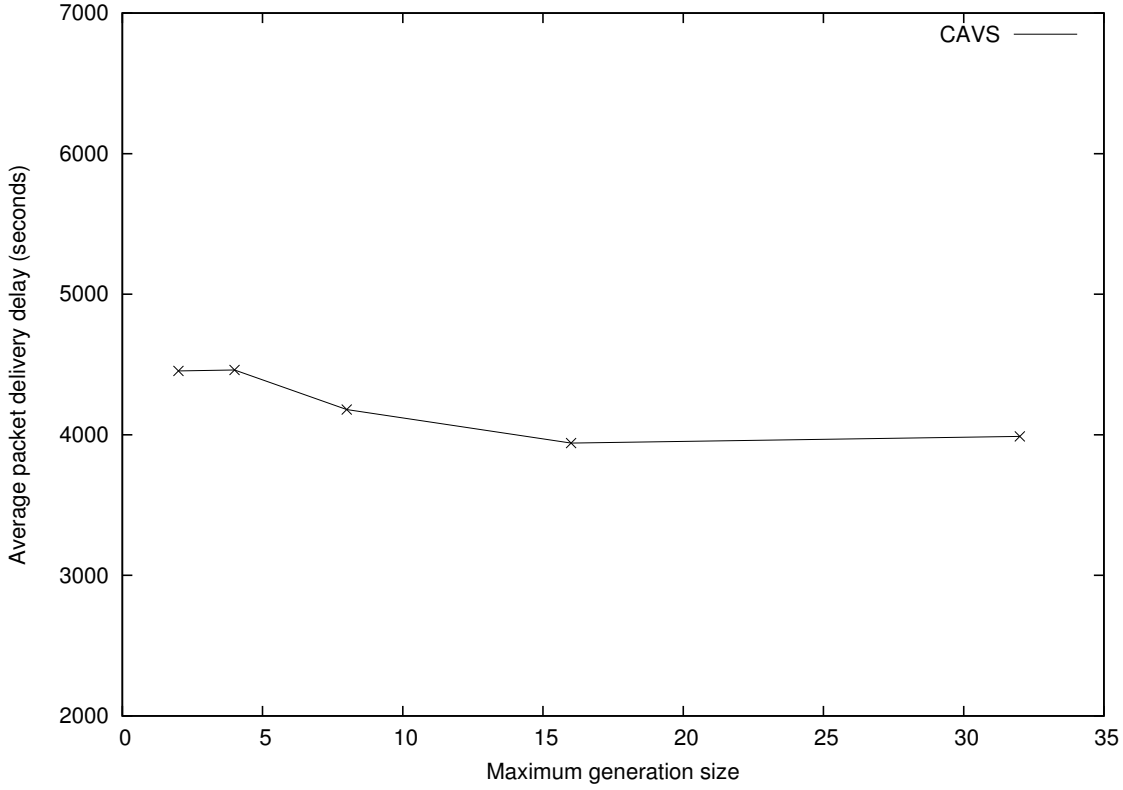

Figure 5.10: Average packet delivery delay vs. Maximum generation size (G)

generation size $G = 16$ approximately.

5.3.1 Modeling Viral Spread as SI (Susceptible Infected) Epidemic Model

An epidemic model is used to describe and analyze the transmission of communicable disease through individuals. Let S be susceptible, I be Infected, β be contact rate, P be infection probability, and N be population size. Then, as per SI (Susceptible Infected) model[70],

$$\begin{aligned} \frac{dS}{dt} &= -\frac{\beta PSI}{N} \\ \frac{dI}{dt} &= \frac{\beta PSI}{N} \end{aligned} \tag{5.2}$$

Our protocol can also be modeled as SI epidemic model. For a packet, S represents nodes which haven't received a packet, β represents average contact rate between nodes, P represents buffering probability, and N represents the number of nodes. Using the model, we have calculated the time to deliver the packet to 95% of nodes with different forwarding probabilities. As shown in Fig. 5.11, the rate of decrease in delivery time with an increase in forwarding factor is high till buffering probability $P_b = 0.3$. Beyond that, with an increase in P_b, the rate of decrease in delivery time is low. The result is in conformance with the results obtained through simulation.

5.4 Conclusion

Many-to-all broadcast in MSN can be useful to a variety of applications in MSN such as message broadcast, news spread, traffic updates etc. It is also useful for routing as many routing protocols require flooding of control messages. We propose Community Aware Viral Spread (CAVS) protocol for MSN.

We simulate CAVS with realistic mobility model (CAHM) which models properties of human mobility and compare it with modified Epidemic routing. Simulation results show that CAVS gives acceptable performance for P_b value as low as 0.3 as compared to $P_b = 1$. Also, with a decrease in P_b, the performance of CAVS gets increasingly better

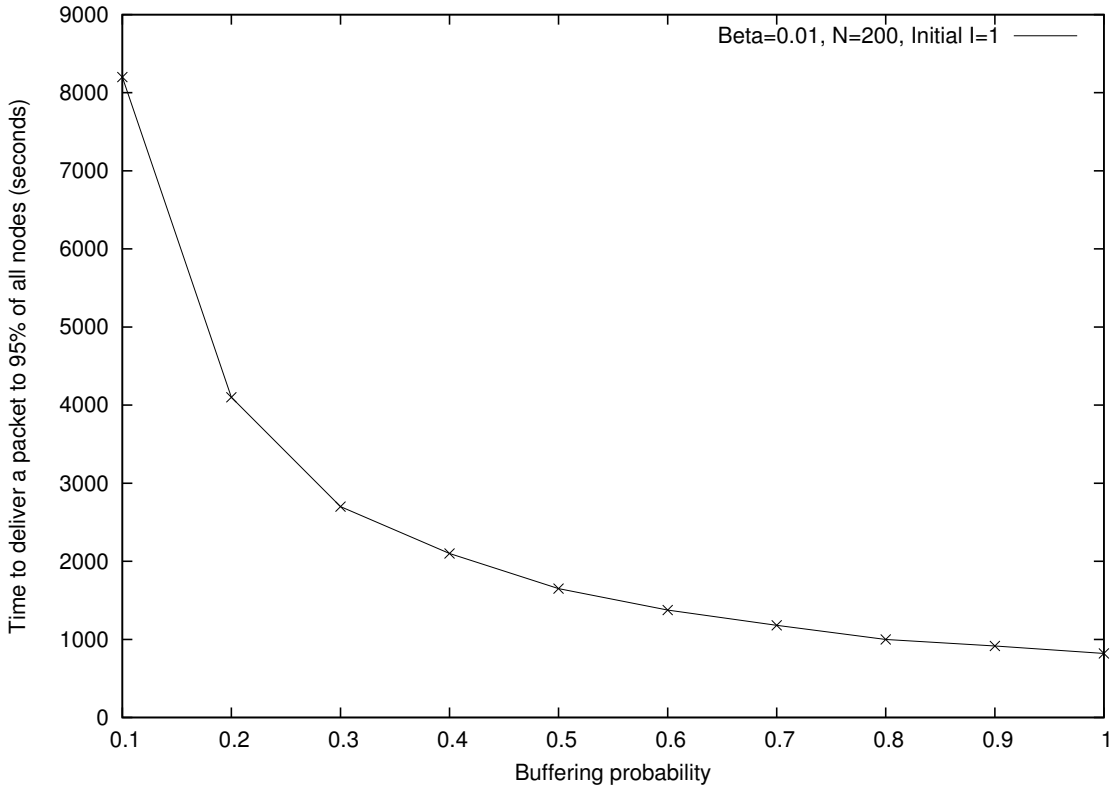

Figure 5.11: Time to deliver a packet to 95% of all nodes vs. Buffering probability (P_b) as per SI epidemic model

than Epidemic routing. For $P_b = 0.1$, packet delivery ratio and average packet delivery delay of CAVS is 122% more and 22% less than Epidemic routing respectively. Starting from $P_b = 0.1$, the rate of increase in packet delivery ratio and the rate of decrease in average packet delivery delay in CAVS and Epidemic routing protocols are high till $P_b = 0.3$. So, we recommend operating the protocol with $P_b = 0.3$ approximately. The large performance difference between CAVS and CAVS (without hubs and gateways) for lower P_b values shows that hub and gateway nodes play very significant role in the performance improvement of CAVS protocol.

Results also show that gateway nodes play greater role in the performance improvement of CAVS as compared to hub nodes and improvement in packet delivery ratio and average packet delivery delay beyond 20% of total nodes as popular nodes is not significant. So, we recommend using 20% of total nodes in the network as hub and gateway nodes. Further, packet delivery ratio is almost independent of the maximum generation size (G). But, there is a 12% improvement in average packet delivery delay with $G = 16$ as compared to average packet delivery delay with $G = 2$. Also, after $G = 16$, it remains

almost constant. So, we recommend setting maximum generation size $G = 16$ approximately. With recommended values of buffering probability (P_b), maximum generation size (G), and percentage of total nodes as popular nodes, packet delivery ratio and average packet delivery delay of CAVS is 58% more and 41% less than Epidemic routing respectively.

We model CAVS protocol as SI (Susceptible Infected) epidemic model and show that optimal buffering probability found through simulation is in conformance with the optimal buffering probability calculated using the model.

Chapter 6

Scalable Micro-blogging in Mobile Social Network

Micro-blogging, particularly Twitter, is very popular among Internet users. Users post small, 140 characters long, messages called tweets. These tweets are about their status, opinions, news, events, emergency situations, traffic updates, advertisements etc. Users tag their tweets based on its content which is called hashtag. Each user receives tweets from users she is following. One can follow a user but not a topic in the system. A user does not get all tweets, with hashtags she is interested in, on her timeline because a huge number of tweets are generated for a hashtag across the Internet. User gets tweets with a hashtag only when she searches for it.

Micro-blogging can be a very promising application for Mobile Social Network (MSN) in which small messages posted by users can be opportunistically pushed in the network without using any infrastructure. In MSN, location-based communities are formed; e.g. students of a same class, workers of a same department etc. Users of these communities have similar interests[41]. Further, co-located users are also interested in temporal events taking place in the vicinity; e.g. a cricket match between teams of two college departments. Because of these spatiotemporal properties of user interests and generated messages, messages can be pushed to users up to a limited distance and for a limited time without overwhelming them. So, users can receive messages in which they are interested without having to identify and follow other users with similar interests. Such an approach can be very much scalable as message filtering is done at the network protocol

level itself as opposed to the conventional approach where it is done at the end points of the network. It also eliminates the need to maintain a large amount of data at server machines. As server farms consume a huge amount of energy[42], it is useful to not to use infrastructure even though the bandwidth requirement of the application is limited.

Based on these observations, we propose a distributed and scalable micro-blogging protocol which exploits community structure and heterogeneous popularity of nodes. For performance analysis of the protocol through simulation, we also propose three novel models for generating user interest profiles synthetically resembling real-world scenario.

In the following section 6.1, we survey relevant literature. We present our protocol in section 6.2. In section 6.3, we propose models to generate user interest profiles synthetically. Section 6.4 discusses simulation results. We conclude the chapter in section 6.5.

6.1 Literature Survey

Content dissemination protocols based on publish/subscribe model are similar to our application. Socio-aware overlay[27] is based on non-overlapping community structure. It takes advantage of only hub nodes but not gateway nodes and actual routing between hub nodes of different communities is left to any standard unicast protocol. Further, modeling of user interest profile is not considered and there is no consideration for scalability. Socialcast[41] and ContentPlace[29] calculate a utility function and based on that decide where to place each message. We rely on hub and gateway nodes for all messages instead of detailed information exchange and calculation for each message.

MoP-2-MoP[71] is a micro-blogging protocol for MSN which focuses on privacy, anonymity, and censorship resistance. It does not aim to deliver messages efficiently and employs flooding to disseminate messages. Allen et al. in [4] have also proposed a micro-blogging protocol for MSN named as 'Uttering'. We compare the performance of our protocol with 'Uttering' protocol as it aims to deliver messages to interested users efficiently which is also our objective. So, we describe 'Uttering' in some detail in the following sub-section.

6.1.1 Overview of 'Uttering' Protocol

In 'Uttering', nodes exchange their users' interest profiles with each other opportunistically. The user interest profile is defined as a set of tags in which a user is interested, along with the level of interest in each tag. Each node identifies set of nodes with interest profiles 'similar' to its own interest profile as friend nodes and set of nodes with frequent contacts as familiar nodes. Friend nodes also exchange aggregate interest profile of their own friends with each other. Each node calculates push profile for each of its friend nodes based on friend node's interest profile and aggregate interest profile of friend node's friends. Each node also calculates community profile which is the aggregation of all push profiles it has calculated for its friends.

In 'Uttering', authors have proposed different forwarding protocol variants. Here, we describe the variant with the best performance as per their results. They call this variant as 'Protocol B'. In this variant, upon an encounter, a message is pushed to a friend node only if in the push profile calculated by the node for the friend node, interest in the tag of the message is non-zero. In the protocol, the maximum number of messages that can be pushed upon each encounter is restricted. So, the next message to be pushed is probabilistically selected using a roulette wheel selection based on the relative interest values stored in the push profile. Similarly, messages are also pushed to a familiar node based on the community profile. To a stranger node, randomly chosen messages are pushed till the maximum message limit is reached.

6.2 The Proposed Scalable Micro-blogging Protocol

When two nodes come in contact, each node pushes to the other node messages with tags in which the other node is interested in. Users can specify their interests or they can be derived from their posts. To increase chances of delivery to interested nodes, all messages of a community are also forwarded to the hub and gateway nodes of the community. Hub nodes help in spreading messages in the community while gateway nodes help in spreading messages to other communities.

Nodes share their interest profiles with community members including hub nodes of the community. Hub and gateway nodes share and accumulate aggregate interest profiles

of communities. Control message overhead of the protocol is limited as control messages are exchanged predominantly only within a community. Further, control messages across communities are restricted by sharing aggregate interest profiles of a limited number of communities with only hub and gateway nodes.

Based on aggregate interest profiles, the threshold distance is estimated up to which interest is relatively high in the message. Message forwarding is restricted based on the threshold distance. Messages in which a neighbour node is not interested are forwarded only if it is a hub or a gateway node. So, the forwarding overhead of the protocol is limited. Hence, the protocol is highly scalable and it scales with community size rather than with the size of the network.

Salient features of our protocol are following.

- We exploit social properties of humans such as overlapping community structure and heterogeneous popularity of nodes for micro-blogging.
- We limit control message overhead such that it is independent of the number of communities and network size.
- We limit forwarding overhead by forwarding messages to only hub and gateway nodes, that too based on distance.

The following sub-sections describe the protocol in detail. We also compare each aspect of the protocol with the relevant part of the 'Uttering' protocol.

6.2.1 Network Model

Nodes in the network form overlapping community structure. Each community is associated with a place (office, home, park etc.). The location of the place is chosen randomly. A node can be part of multiple communities. A node moves between different locations within a place when it is in a community. The number of different locations in a community that a node visits depends on its popularity in the community. A node visits places associated with those communities in which it is a member. A node moves within a place more frequently while between places, it moves less frequently. The nodes in the network find such overlapping community structure using Distributed Overlapping Community Detection (DOCD) mechanism proposed in chapter 3. The network is typically sparse

and there is generally no contemporaneous path between source and destination. Due to this, we use opportunistic forwarding where packets are exchanged opportunistically when two nodes come within communication range of each other.

'Uttering' does not explicitly find community structure. But in 'Uttering', each node maintains its community as a set of its friend nodes as well as friend nodes' friends; i.e. for a node, the community is set of nodes having similar interests. So, each node has its own notion of community. Further, the community is found based on interest similarity instead of location similarity.

6.2.2 Hub and Gateway Nodes

Locally popular (hub) nodes of a community come in contact with more number of nodes in the community than rest of the nodes of the community. Similarly, globally popular (gateway) nodes of a community visit multiple communities and come in contact with more number of nodes in the entire network than rest of the nodes of the community. We use methods described in chapter 4 to identify hub and gateway nodes from overlapping community structure itself without doing message flooding or forwarding of accumulated encounter information from other nodes. These methods order nodes of a community based on their suitability as hub or gateway node. From the ordered list of candidate hub nodes, we choose a fraction of most suitable nodes as hub nodes of the community. Similarly from the ordered list of candidate gateway nodes, a fraction of most suitable nodes are chosen as gateway nodes of the community.

'Uttering' does not identify hub and gateway nodes in the network. Rather, it relies on friend nodes, familiar nodes and stranger nodes to forward messages in the network so that messages can reach interested nodes.

6.2.3 Control Messaging

Let set of tags in which node n is interested be $\mathbf{S}^n$. Then, user interest profile for node n is defined as $\mathbf{P}^n = \{P_i^n : i \in \mathbf{S}^n, 0 < P_i^n \leq 1\}$ where P_i^n is level of interest of node n in tag i. Also, set of nodes of community J interested in tag i is defined as $\mathbf{C}_i^J = \{n : n \in J, i \in \mathbf{S}^n\}$. Further, aggregate interest profile of community J is defined

as $\mathbf{I}^J = \{I_i^J : i \in \bigcup_{n \in J} \mathbf{S}^n\}$ where

$$I_i^J = \frac{\sum_{n \in \mathbf{C}_i^J} P_i^n}{\left|\mathbf{C}_i^J\right|} \tag{6.1}$$

When two nodes of a community encounter each other the first time, they exchange their user interest profiles. Upon subsequent encounters, they exchange change in interest profile since last encounter. If a node encounters a hub node, it also sends accumulated interest profiles of previously encountered nodes to the hub node. Hub nodes calculate average interest in each tag along with the number of nodes of the community interested in the tag. It is called aggregate interest profile of the community.

Hub and gateway nodes of the network exchange and accumulate aggregate interest profiles of communities when they encounter each other. A substantial change in the aggregate interest profile of a community is also propagated by community hub nodes to other hub and gateway nodes. To keep amount of state information maintained at hub and gateway nodes independent of number of communities and network size, for each tag, aggregate interest is stored only for given number of communities in which interest for the tag is relatively high. This also reduces the amount of control information exchanged between hub and gateway nodes of the network.

In 'Uttering', nodes exchange their user interest profile when they encounter the first time. If two nodes have similar interests then they identify each other as friends. Upon each encounter, friend nodes also exchange aggregate interest profiles of their own friends. In 'Uttering', each node exchanges its interest profile with all the nodes it encounters from the entire network. In our protocol, only community members exchange individual interest profiles. Further, in 'Uttering', each node transmits aggregate interest profile of its friends to each of its friends and it also needs to maintain aggregate interest profiles received from each of its friends. In our protocol, only single aggregate interest profile per community is maintained and exchanged between hub and gateway nodes of the network only. Further, for each tag, hub and gateway nodes store aggregate interest of only given number of communities in which interest for the tag is relatively high. So, control message overhead in terms of storage and transmission is substantially less in our protocol as compared to 'Uttering'.

6.2.4 Threshold Distance Calculation

Threshold distance (D) for a tag from a community is the distance up to which messages originated from the community with the tag are forwarded to gateway nodes of the network with probability 1. Hub nodes of each community calculate threshold distance for each tag of the community from accumulated aggregate interest profiles.

Each hub node calculates distances from its community to all other communities and sort communities based on their distances in increasing order. Let $N_i^J = \left|\mathbf{C}_i^J\right|$ be number of nodes of community J interested in tag i, D_i^J be threshold distance for tag i from community J and M be number of communities. Then,

$$F_i^L = \frac{\sum_{K=1}^{K=L} I_i^K * N_i^K}{\sum_{K=1}^{K=M} I_i^K * N_i^K}, L = 1, 2, ..., M \tag{6.2}$$

Here $L = 1$ represents the community of the hub node, $L = 2$ represents the nearest community and so on. F_i^L represents the ratio of aggregate interest in tag i till community L to total aggregate interest in tag i. Then, threshold distance (D_i^J) is the distance of community L such that $F_i^L > I_{th}$ where I_{th} represents threshold interest which is a protocol parameter. So, the threshold distance (D) covers communities such that their normalized cumulative interest in a tag is greater than threshold interest (I_{th}) which is between 0 and 1.

'Uttering' does not estimate distance up to which interest in the message is high. Instead, at the time of the creation of the message only, distance limit for the message is specified.

6.2.5 Probabilistic Forwarding

Actual messages are forwarded opportunistically when a node comes in contact with another node. If the neighbour node is neither hub nor gateway of the current community then only messages of interest to the neighbour node are forwarded to it. Else, all messages of the current community are forwarded. If a node is a gateway node of any community and the neighbour is either hub or gateway node of current community then messages of other communities are forwarded as follows: If the distance of the current node from the community J of a message with tag i is less than threshold distance D_i^J, then the message is forwarded. Else, it is forwarded probabilistically where probability decreases

with distance. This forwarding rule restricts forwarding of the message based on interest in the message, thereby making the protocol efficient and scalable.

We assume that a node will be able to send all messages needed to be sent as per protocol to a neighbour node upon each encounter based on the fact that in micro-blogging, message size is very small and average contact period is sufficient to send all messages over Wifi. We also do not consider limited buffer case because of small message size.

In 'Uttering', each node relies on its own friend nodes and familiar nodes for efficient message forwarding. A node also forwards randomly selected messages to a stranger node. These nodes are not necessarily popular nodes of the network. Instead, our protocol relies on hub nodes for efficient forwarding within a community and gateway nodes for efficient forwarding between communities. As these are popular nodes of the network, they are far more effective for message forwarding then forwarding nodes of 'Uttering'. Further, in 'Uttering', a message may be pushed to different sets of friend nodes and familiar nodes of different nodes. So, forwarding overhead of the protocol is quite high. In our protocol, we forward a message to a node, even though it is not interested in it, only if the node is either hub or gateway node. Further, we can precisely control the number of hub and gateway nodes in the network. So, the forwarding overhead of our protocol is very much limited.

6.3 User Interest Profile Generation

From the analysis of Twitter data, it is shown in [72] that a large number of users are interested in a small number of tags. Further, users with similar interests are co-located [41] and interest in a topic decreases with distance. Based on these properties, we propose three novel models to synthetically generate user interest profiles to evaluate the performance of micro-blogging protocols in MSN through simulation.

A user is interested in a subset of all tags with different interest levels in each tag. List of tags along with interest levels in each tag is called user interest profile. A user generates messages as per her interest profile and is also interested in receiving messages as per this profile. We propose following three models for generating user interest profiles synthetically.

6.3.1 Random Model

A given number of tags are randomly assigned to each user irrespective of its community membership from the set of all tags such that the number of users per tag follows Zipf[73] distribution. So, users interested in a tag are spread across different communities existing at different locations. The model represents interests which are not local.

User interest levels in tags assigned to her are Zipf distributed with a random exponent. In this model, we do not maintain any correlation between the level of interest of users in a tag with the number of users interested in the tag.

6.3.2 Community-based Model

In this model, users interested in a tag are predominantly part of few communities only and only few users from remaining communities are interested in the tag. The model represents interests which are local. It is more realistic than the one proposed in [4] where the set of tags of each community is independent of other sets.

To generate user interest profiles based on this model, we shuffle set of all tags with different random seeds for each community. Then, tags to individual users in a community are assigned using Zipf random variate from the shuffled sequence of tags for the community. As Zipf returns lower index values with high probability, tags at lower index values in the shuffled sequence are assigned to a high number of users in a community. As different, but overlapping, sets of tags are at lower index values in shuffled sequences of different communities, this method generates desired profiles.

6.3.3 Distance Aware Community-based Model

It is an extension of the community-based model. Let the density be the percentage of users in a community interested in a tag. In this model, for each tag in a community, density in other communities should be either proportional to the distance from the community or more than that. If it is less than required in a community then an appropriate number of additional users are assigned the tag in that community. For brevity, we subsequently call the model as the distance-based model.

Table 6.1: Simulation parameters

Number of nodes	200
Simulation area	41000 meters x 41000 meters
Number of total tags	400
Number of tags of each node	30
Zipf exponent for tag popularity for random model	0.5
Zipf exponent for tag popularity for other two models	2
Zipf exponent for user interest levels in tags	Randomly chosen between 0 and 2
Threshold interest I_{th} for threshold distance calculation	1
Zipf exponent for density in distance-based model	1
Percentage nodes of each community as hubs	20%
Percentage nodes of each community as gateways	20%
Communication range	40 meters
Message size	154 Bytes
Inter-packet arrival time	140 seconds
TTL	400 minutes

6.4 Simulation Results

To evaluate the performance, we have implemented our micro-blogging protocol in ONE simulator. We evaluate our protocol using Community Aware Heterogeneous Human Mobility (CAHM) model proposed in chapter 2 which moves nodes in the simulation as per our network model. For comparison, we have also implemented 'Uttering' protocol proposed in [4] for micro-blogging in MSN. We measure and compare 'efficiency' and 'spread index' of our protocol with 'Uttering'. Efficiency is defined as the ratio of total number of useful messages received by all nodes to the total number of messages transmitted by all nodes. Spread index is defined as the ratio of total number of users who have received useful messages to the total number of users who were interested in those messages. Spread index and efficiency measures are same as 'recall' and 'precision' measures used for the performance measurement of 'Uttering' in [4]. Simulation parameters are as per Table 6.1 unless otherwise mentioned. All readings are taken once network reached steady state. With a random seed, CAHM generated 14 overlapping communities for 200 nodes. In 'Uttering', we restrict the number of messages to be sent per encounter to 10 messages as done in [4].

As shown in Table 6.2, our protocol significantly outperforms 'Uttering' for all types of user interest profiles. Spread index of our protocol is better than existing protocol

Table 6.2: Comparison with 'Uttering'

Model	Spread index		Efficiency	
	Our Approach	'Uttering'	Our Approach	'Uttering'
Random	0.71	0.29	0.30	0.13
Community-based	0.82	0.50	0.48	0.31
Distance-based	0.81	0.66	0.57	0.32

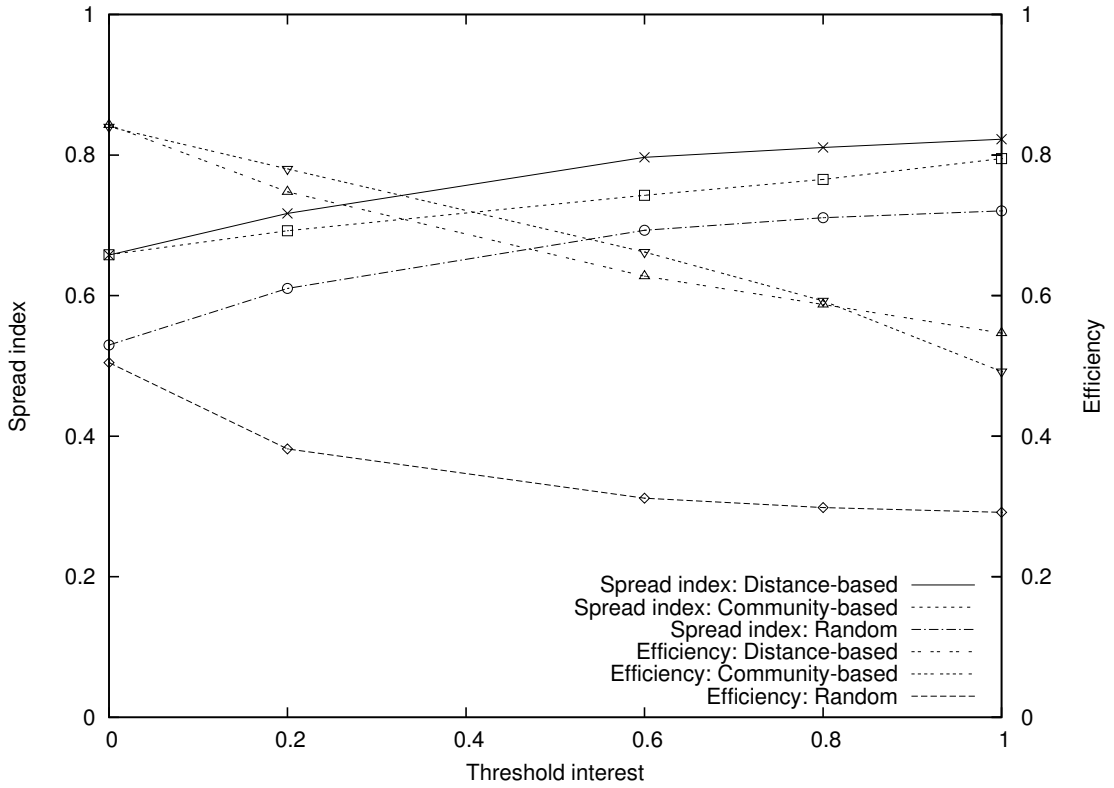

Figure 6.1: Spread index and Efficiency vs. Threshold interest (I_{th})

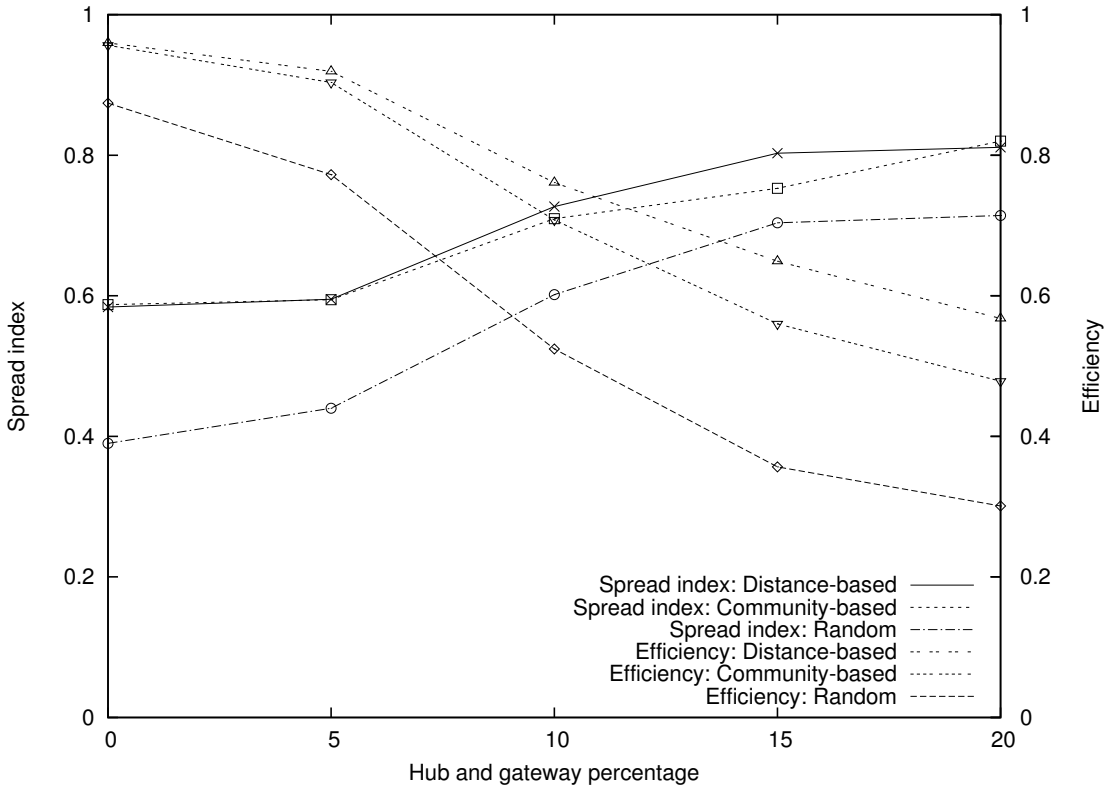

Figure 6.2: Spread index and Efficiency vs. Hub and gateway percentage

(Uttering) by 18-59% and efficiency is better than 'Uttering' by 35-56% for different user interest profile models. Spread index of our protocol is better because 'Uttering' relies on other ordinary non-interested nodes to reach interested nodes while we rely on hub and gateway nodes for the same. Efficiency of our protocol is better because we forward messages to neighbour node, even if it is not interested in the message, only if it is hub or gateway node and hub and gateway nodes are only small percentage of total nodes.

To see the effect of the threshold distance, we disable probabilistic forwarding. Variants with and without probabilistic forwarding are referred as 'Probabilistic' and 'Absolute' variants respectively. As shown in Fig. 6.1, as threshold interest increases in 'Absolute' variant, spread index increases and efficiency decreases as expected for all types of user interest profiles. Further, the performance is far better for community-based and distance-based interest profiles as compared to random interest profile. It is because, in these profiles, nodes interested in a tag are co-located while, in random interest profile, nodes interested in a tag are spread across communities.

As shown in Fig. 6.2, as the percentage of nodes acting as hub and gateway nodes increases, spread index increases and efficiency decreases substantially. So, depending on

the network traffic, number of hub and gateway nodes should be chosen. Further, with less number of hub and gateway nodes, load balancing can be done by rotating the hub and gateway responsibility among eligible nodes. The result is for 'Absolute' variant but the behaviour remains the same for 'Probabilistic' variant also.

Fig. 6.3 shows the effect of storing aggregate interests in a tag of only a limited number of communities at hub and gateway nodes, on the performance. As seen in the figure, for a tag, out of 14 possible entries for 14 communities, entries of only 2 communities which have highest interest in the tag are sufficient. As for more than 2 entries, the performance does not change; i.e. the amount of control information needed in our protocol is independent of the number of communities and network size.

Fig. 6.4 shows comparison of 'Probabilistic' and 'Absolute' variants with different threshold interest values for distance-based user interest profile model. For lower threshold interest values, 'Probabilistic' variant has better spread index but less efficiency as compared to 'Absolute' variant. The difference between spread indexes as well as efficiencies of both variants is negligible for higher threshold interest values. It is expected as, with increase in threshold interest, number of interested nodes not covered by 'Absolute' variant (which can be covered by 'Probabilistic' variant only) decreases. For community-based and random user interest profile models also, the results are similar.

6.5 Conclusion

Micro-blogging can be a very promising application for MSN because of spatiotemporal properties of user interests and generated messages. Users can follow their interests and receive messages of their interests without getting overwhelmed, instead of having to identify and follow users with similar interests. We propose a micro-blogging protocol for MSN exploiting its overlapping community structure and heterogeneously popular nodes. The protocol restricts message forwarding based on distance and keeps a constant amount of state information. The protocol is scalable and efficient as message filtering is done at the network protocol level itself as opposed to the conventional approach where it is done at the end points of the network. It also eliminates the need to maintain a large amount of data at server machines. As server farms consume a huge amount of energy, it is useful to not to use infrastructure even though the bandwidth requirement of the

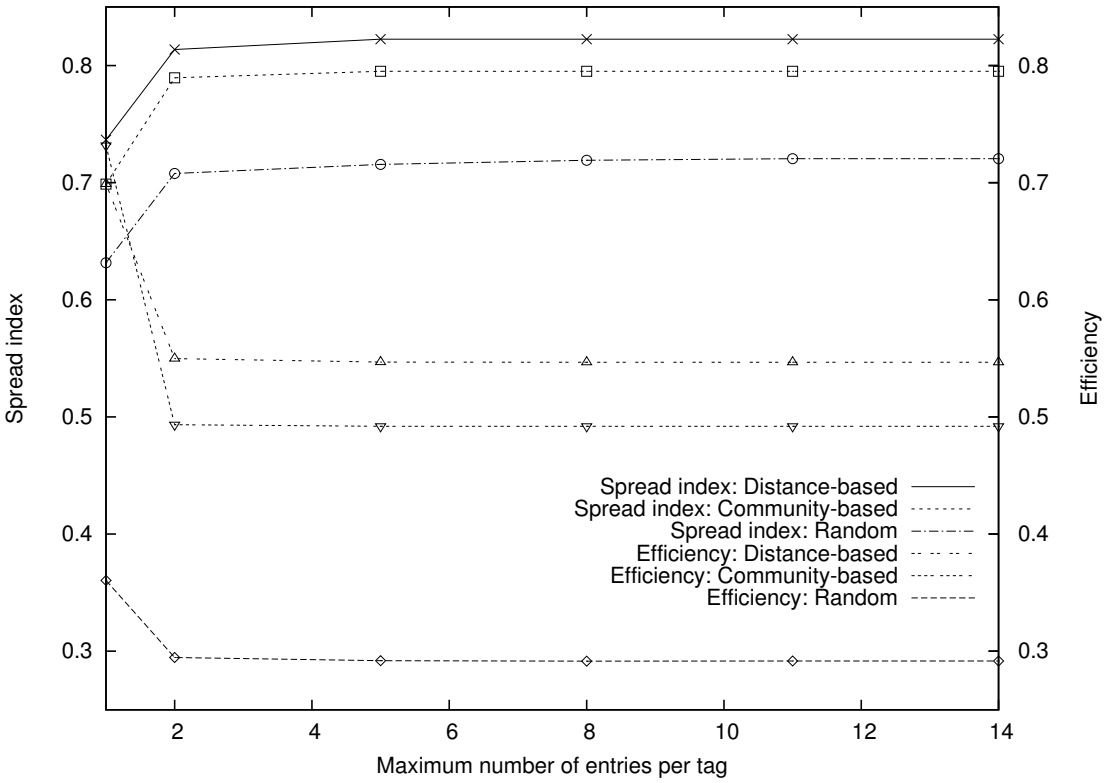

Figure 6.3: Effect of a limited number of entries per tag on the performance

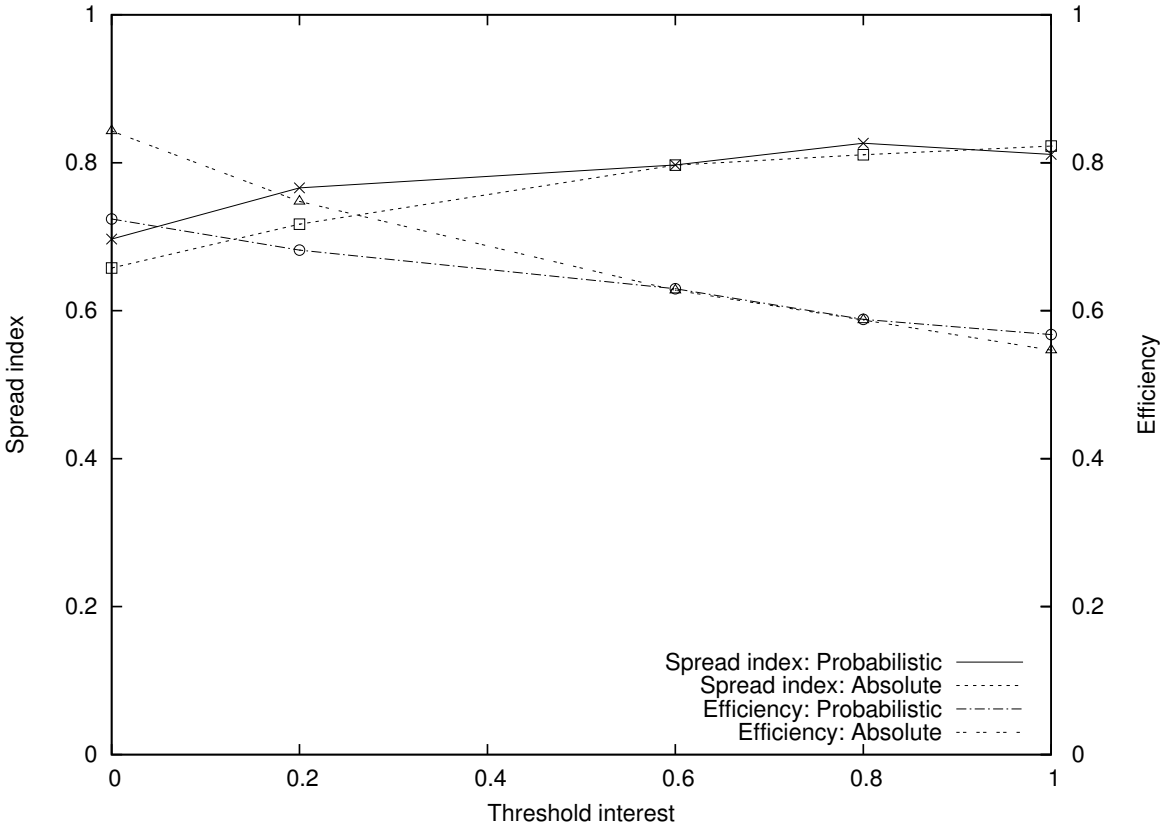

Figure 6.4: Comparison: 'Probabilistic' and 'Absolute' variants

application is limited. We also propose three models to generate user interest profiles synthetically out of which Distance Aware Community-based model is most realistic. Simulation results show that spread index of our protocol is better than existing protocol (Uttering) by 18-59% and efficiency is better than 'Uttering' by 35-56% for different user interest profile models. Simulation results also show that keeping only a constant amount of state information does not degrade the performance of our protocol.

Chapter 7

Conclusion

Peer-to-peer opportunistic communication between mobile devices carried by humans without using any infrastructure is largely unexploited. This network paradigm is known as Mobile Social Network (MSN). Message broadcast, news spread, traffic updates, micro-blogging, and peer-to-peer file sharing are some of the applications which can run on such type of networks. In this thesis, we first propose a realistic mobility model for MSN for reliable performance analysis through simulation. Then, we introduce mechanisms for detecting overlapping community structure and for identification of hub and gateway nodes. Finally, we propose protocols for message broadcast and micro-blogging applications in MSN.

To analyze the performance of routing protocols aiming to exploit human movement behaviour through simulation, it is essential to design realistic mobility models which can mimic human mobility patterns as closely as possible. Several real-world mobility traces have established that human mobility is not random. So, traditional mobility models such as Random Way Point (RWP) and Brownian Motion (BM) should not be used for reliable analysis of protocol performance in MSN through simulation. Various characteristics of human mobility are derived from mobility traces and from social network theory in the literature. No existing mobility model, except HHW, generates community structure synthetically incorporating these characteristics and without using Social Network Models such as Caveman model. We propose Community Aware Heterogeneous Human Mobility (CAHM) model with four modifications in HHW: incorporation of Levy walk nature of human mobility, treatment of number of cells and number of nodes in a com-

munity as separate parameters, calculation of speed based on distance to be traveled and power-law pause time. Simulation results demonstrate that CAHM successfully generates flight lengths with power-law distribution while in HHW flight lengths are uniformly distributed. Further, movement of individuals in CAHM is as per rational human behaviour of preference of nearby locations over far-away locations while in HHW it is not. The results also establish that CAHM generates desired heterogeneous local popularity of nodes while HHW generates too many highly popular nodes.

We also analyze the effect of mobility models on the performance of routing protocols in MSN. Simulation results confirm that exploiting community structure and heterogeneous node popularity significantly improves performance. The results also show that due to less realistic mobility models, simulation significantly underestimates the performance of protocols. The packet delivery ratio of Epidemic routing and BUBBLE Rap routing increases by 21-28% with CAHM as compared to HHW in our simulation setup. The result shows that additional properties of human mobility incorporated in our mobility model such as heterogeneous popularity of nodes as observed in mobility traces, Levy walk nature of human mobility, speed as a function of distance, and power-law pause time have significant impact on the performance of routing protocols.

Routing protocols in MSN can exploit overlapping community structure formed by humans for efficient forwarding. This structure can also be used to identify hub and gateway nodes of a community without doing message flooding. Nodes in MSN need to detect overlapping community structure in a decentralized manner. There is no mechanism available in the literature for the same. We modify existing distributed algorithm SIMPLE to detect overlapping community structure and propose Distributed Overlapping Community Detection (DOCD) mechanism. Simulation results show that DOCD detects overlapping community structure with 75-80% accuracy. Further, it is evident from the results that the performance of DOCD is good if the value of the initial threshold distance (d_{th}) is kept low, and it is not sensitive to the familiarity threshold parameter.

Our analysis of overlapping community structure establishes that small communities are transient. As per simulation results, the threshold for the same is around 10% of total number of nodes in the network. Further, a higher fraction of hub and gateway nodes remain present in larger communities with less standard deviation as compared to smaller communities. Also, the fraction of gateway nodes present in a community is

much lower than the fraction of hub nodes present. These results give important insights for designing better forwarding protocols for MSN.

Hub and gateway nodes of a community can play very important role in the efficient dissemination of information in MSN. Existing methods to detect hub and gateway nodes require either flooding of messages or forwarding of not only node's encounter information but also accumulated encounter information from other nodes. So, the performance of these methods does not scale with network size or community size. We identify hub and gateway nodes from the overlapping community structure itself without doing message flooding or forwarding of accumulated encounter information from other nodes. We propose Markov chain-based methods to identify hub and gateway nodes of a community. The methods involve exchange of independently calculated popularity values of a node with community members only. So, the communication cost (as well as the computational cost) of the method is $\mathcal{O}(C^3)$ only, where C denotes maximum community size; i.e. the performance of Markov chain-based methods scales with the number of nodes in the network. While, for the baseline method, the communication cost (as well as the computational cost) is $\mathcal{O}(n^3)$, where n is the number of nodes in the network. To validate Markov chain-based methods, we also propose two simulation-based approaches to identify hub nodes and another two simulation-based approaches to identify gateway nodes. We compare ordered lists of hub and gateway nodes generated by Markov chain-based methods with ordered lists generated by these methods. Simulation results show that these ordered lists are highly correlated, i.e. Markov chain-based methods correctly identify hub and gateway nodes.

Many-to-all broadcast in MSN can be useful to a variety of applications in MSN such as message broadcast, news spread, traffic updates etc. It is also useful for routing as many routing protocols require flooding of control messages. We propose Community Aware Viral Spread (CAVS) protocol for MSN. We simulate CAVS with realistic mobility model (CAHM) which models properties of human mobility and compare it with modified Epidemic routing. Simulation results show that CAVS gives acceptable performance for P_b value as low as 0.3 as compared to $P_b = 1$. Also, with a decrease in P_b, the performance of CAVS gets increasingly better than Epidemic routing. For $P_b = 0.1$, packet delivery ratio and average packet delivery delay of CAVS is 122% more and 22% less than Epidemic routing respectively. Starting from $P_b = 0.1$, the rate of increase in packet delivery

ratio and the rate of decrease in average packet delivery delay in CAVS and Epidemic routing protocols are high till $P_b = 0.3$. So, we recommend operating the protocol with $P_b = 0.3$ approximately. The large performance difference between CAVS and CAVS (without hubs and gateways) for lower P_b values shows that hub and gateway nodes play very significant role in the performance improvement of CAVS protocol. Results also show that gateway nodes play greater role in the performance improvement of CAVS as compared to hub nodes and improvement in packet delivery ratio and average packet delivery delay beyond 20% of total nodes as popular nodes is not significant. So, we recommend using 20% of total nodes in the network as hub and gateway nodes. Further, packet delivery ratio is almost independent of the maximum generation size (G). But, there is a 12% improvement in average packet delivery delay with $G = 16$ as compared to average packet delivery delay with $G = 2$. Also, after $G = 16$, it remains almost constant. So, we recommend setting maximum generation size $G = 16$ approximately. With recommended values of buffering probability (P_b), maximum generation size (G), and percentage of total nodes as popular nodes, packet delivery ratio and average packet delivery delay of CAVS is 58% more and 41% less than Epidemic routing respectively. We model CAVS protocol as SI (Susceptible Infected) epidemic model and show that optimal buffering probability found through simulation is in conformance with the optimal buffering probability calculated using the model.

Micro-blogging can be a very promising application for MSN because of spatiotemporal properties of user interests and generated messages. Users can follow their interests and receive messages of their interests without getting overwhelmed, instead of having to identify and follow users with similar interests. We propose a micro-blogging protocol for MSN exploiting its overlapping community structure and heterogeneously popular nodes. The protocol restricts message forwarding based on distance and keeps a constant amount of state information. The protocol is scalable and efficient as message filtering is done at the network protocol level itself as opposed to the conventional approach where it is done at the end points of the network. It also eliminates the need to maintain a large amount of data at server machines. As server farms consume a huge amount of energy, it is useful to not to use infrastructure even though the bandwidth requirement of the application is limited. We also propose three models to generate user interest profiles synthetically out of which Distance Aware Community-based model is most realistic. Simulation re-

sults show that spread index of our protocol is better than existing protocol (Uttering) by 18-59% and efficiency is better than 'Uttering' by 35-56% for different user interest profile models. Simulation results also show that keeping only a constant amount of state information does not degrade the performance of our protocol.

Chapter 8

Future Work

Successful peer-to-peer communication in Mobile Social Network (MSN) requires that devices in the network must cooperate in forwarding messages of other devices. But, as resources such as computing power, bandwidth, and battery power are limited, there is enough motivation for a device in MSN to be selfish. Even if we assume that a non-selfish application is supplied by single developer so that no device can act selfishly by modifying the protocol, it is possible for a device to be selfish by switching off communication interfaces like Wifi, Bluetooth, and data connection when device does not need them. We intend to explore possibilities to enforce cooperation in our viral spread and micro-blogging protocols when communication interfaces are switched off selfishly. Further, in our viral spread and micro-blogging protocols, we identify some of the nodes as hub and gateway nodes based on their local and global popularity respectively. These devices do more work as compared to other nodes for efficient forwarding. We intend to incorporate incentive mechanisms in our protocols for these nodes.

Bibliography

[1] S. Yang, X. Yang, C. Zhang, and E. Spyrou, "Using social network theory for modeling human mobility," *Network, IEEE*, vol. 24, no. 5, pp. 6–13, 2010.

[2] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*. ACM, 2007, p. 7.

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

[4] S. Allen, M. Chorley, G. Colombo, E. Jaho, M. Karaliopoulos, I. Stavrakakis, and R. Whitaker, "Exploiting user interest similarity and social links for micro-blog forwarding in mobile opportunistic networks," *Pervasive and Mobile Computing*, 2011.

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

[6] B. Han, P. Hui, V. Kumar, M. V. Marathe, G. Pei, and A. Srinivasan, "Cellular traffic offloading through opportunistic communications: a case study," in *Proceedings of the 5th ACM workshop on Challenged networks*. ACM, 2010, pp. 31–38.

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

[8] C. Boldrini, M. Conti, and A. Passarella, "Impact of social mobility on routing protocols for opportunistic networks," in *World of Wireless, Mobile and Multimedia*

Networks, 2007. WoWMoM 2007. IEEE International Symposium on a. IEEE, 2007, pp. 1–6.

[9] M. E. Newman, "The structure and function of complex networks," *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[10] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[11] M. Musolesi and C. Mascolo, "Designing mobility models based on social network theory," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 3, pp. 59–70, 2007.

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

[13] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Personal and ubiquitous computing*, vol. 10, no. 4, pp. 255–268, 2006.

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

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

[16] E. Hyytiä, H. Koskinen, P. Lassila, A. Penttinen, J. Roszik, and J. Virtamo, "Random waypoint model in wireless networks," *Networks and Algorithms: complexity in Physics and Computer Science, Helsinki*, 2005.

[17] R. Groenevelt, E. Altman, and P. Nain, "Relaying in mobile ad hoc networks: the brownian motion mobility model," *Wireless Networks*, vol. 12, no. 5, pp. 561 – 571, September 2006.

[18] W.-J. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy, "Modeling spatial and temporal dependencies of user mobility in wireless mobile networks," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 5, pp. 1564–1577, 2009.

[19] A. Mei and J. Stefa, "Swim: A simple model to generate small mobile worlds," in *INFOCOM 2009*. IEEE, 2009, pp. 2106–2113.

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

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

[22] Watts and J. Duncan, *Small worlds: the dynamics of networks between order and randomness*. Princeton university press, 1999.

[23] E. Borgia, M. Conti, and A. Passarella, "Autonomic detection of dynamic social communities in opportunistic networks," in *Ad Hoc Networking Workshop (Med-Hoc-Net), 2011 The 10th IFIP Annual Mediterranean*. IEEE, 2011, pp. 142–149.

[24] M. Orlinski and N. Filer, "The rise and fall of spatio-temporal clusters in mobile ad hoc networks," *Ad Hoc Networks*, vol. 11, no. 5, pp. 1641–1654, 2013.

[25] M. J. Williams, R. M. Whitaker, and S. M. Allen, "Decentralised detection of periodic encounter communities in opportunistic networks," *Ad Hoc Networks*, vol. 10, no. 8, pp. 1544–1556, 2012.

[26] K. Wehmuth and A. Ziviani, "Distributed assessment of the closeness centrality ranking in complex networks," in *Proceedings of the Fourth Annual Workshop on Simplifying Complex Networks for Practitioners*. ACM, 2012, pp. 43–48.

[27] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft, "A socio-aware overlay for publish/subscribe communication in delay tolerant networks," in *Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*. ACM, 2007, pp. 225–234.

[28] F. Li and J. Wu, "Localcom: a community-based epidemic forwarding scheme in disruption-tolerant networks," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON'09. 6th Annual IEEE Communications Society Conference on.* IEEE, 2009, pp. 1–9.

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

[30] M. Stabeler, C. Lee, G. Williamson, and P. Cunningham, "Using hierarchical community structure to improve community-based message routing," in *The Social Mobile Web*, 2011.

[31] L. Zhao, F. Li, C. Zhang, and Y. Wang, "Routing with multi-level social groups in mobile opportunistic networks," in *Global Communications Conference (GLOBECOM), 2012 IEEE.* IEEE, 2012, pp. 5290–5295.

[32] T. Vogelsang, "Understanding the energy consumption of dynamic random access memories," in *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture.* IEEE Computer Society, 2010, pp. 363–374.

[33] R. R. Maiti, N. Ganguly, and A. Gupta, "Epidemic broadcasting in dtns using directional antenna," in *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on.* IEEE, 2012, pp. 1–2.

[34] S. Yusuke, Y. Teranishi, K. Harumoto, S. Takeuchi, and S. Nishio, "A broadcast method based on estimation and preservation of stable links in delay tolerant networks," in *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on.* IEEE, 2011, pp. 427–432.

[35] H. Gong and J. Kim, "A prioritization-based application-oriented broadcast protocol for delay-tolerant networks," in *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE.* IEEE, 2009, pp. 1–6.

[36] F.-Y. Lee, S.-I. Sou, and P. Lin, "Social-based broadcasting for delay tolerant network," in *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*. IEEE, 2013, pp. 2250–2254.

[37] E. Yoneki, P. Hui, and J. Crowcroft, "Wireless epidemic spread in dynamic human networks," in *Bio-Inspired Computing and Communication*. Springer, 2008, pp. 116–132.

[38] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, pp. 1204–1216, 2000.

[39] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "Efficient broadcasting using network coding," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 2, pp. 450–463, 2008.

[40] J. Widmer and J.-Y. Le Boudec, "Network coding for efficient communication in extreme networks," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*. ACM, 2005, pp. 284–291.

[41] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco, "Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks," *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 5, pp. 748–760, 2008.

[42] G. Fettweis and E. Zimmermann, "Ict energy consumption-trends and challenges," in *Proceedings of the 11th International Symposium on Wireless Personal Multimedia Communications*, vol. 2, no. 4, 2008, p. 6.

[43] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *INFOCOM 2003. twenty-second annual joint conference of the IEEE computer and communications. IEEE societies*, vol. 2. IEEE, 2003, pp. 1312–1321.

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

[45] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," *Computer Networks*, vol. 52, no. 14, pp. 2690–2712, 2008.

[46] L. Vu, Q. Do, and K. Nahrstedt, "Jyotish: Constructive approach for context predictions of people movement from joint wifi/bluetooth trace," *Pervasive and Mobile Computing*, vol. 7, no. 6, pp. 690–704, 2011.

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

[48] F. Ekman, A. Keränen, J. Karvo, and J. Ott, "Working day movement model," in *Proceedings of the 1st ACM SIGMOBILE workshop on Mobility models*. ACM, 2008, pp. 33–40.

[49] C. Zhao, M. L. Sichitiu, and I. Rhee, "N-body: A social mobility model with support for larger populations," *Ad Hoc Networks*, vol. 25, pp. 185–196, 2015.

[50] D. Karamshuk, C. Boldrini, M. Conti, and A. Passarella, "Spot: Representing the social, spatial, and temporal dimensions of human mobility with a unifying framework," *Pervasive and Mobile Computing*, vol. 11, pp. 19–40, 2014.

[51] P. Pirozmand, G. Wu, B. Jedari, and F. Xia, "Human mobility in opportunistic networks: Characteristics, models and prediction methods," *Journal of Network and Computer Applications*, vol. 42, pp. 45–58, 2014.

[52] A. Keränen, J. Ott, and T. Kärkkäinen, "The one simulator for dtn protocol evaluation," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009, p. 55.

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

[54] J. Á. Bitsch Link, N. Viol, A. Goliath, and K. Wehrle, "Simbetage: utilizing temporal changes in social networks for pocket switched networks," in *Proceedings of the 1st ACM workshop on User-provided networking: challenges and opportunities*. ACM, 2009, pp. 13–18.

[55] S. Gaito, E. Pagani, and G. P. Rossi, "Strangers help friends to communicate in opportunistic networks," *Computer Networks*, vol. 55, no. 2, pp. 374–385, 2011.

[56] C. Boldrini, M. Conti, F. Delmastro, and A. Passarella, "Context-and social-aware middleware for opportunistic networks," *Journal of Network and Computer Applications*, vol. 33, no. 5, pp. 525–541, 2010.

[57] C. Boldrini, M. Conti, and A. Passarella, "Exploiting users social relations to forward data in opportunistic networks: The hibop solution," *Pervasive and Mobile Computing*, vol. 4, no. 5, pp. 633–657, 2008.

[58] S. Wang, M. Liu, X. Cheng, Z. Li, J. Huang, and B. Chen, "Hero–a home based routing in pocket switched networks," in *Wireless Algorithms, Systems, and Applications*. Springer, 2012, pp. 20–30.

[59] W.-J. Hsu, D. Dutta, and A. Helmy, "Csi: A paradigm for behavior-oriented profile-cast services in mobile networks," *Ad Hoc Networks*, vol. 10, no. 8, pp. 1586–1602, 2012.

[60] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[61] P. Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura," *Bulletin del la Société Vaudoise des Sciences Naturelles*, vol. 37, pp. 547–579, 1901.

[62] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: an open source software for exploring and manipulating networks," in *ICWSM*, 2009, pp. 361–362.

[63] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[64] R. Lambiotte, J.-C. Delvenne, and M. Barahona, "Laplacian dynamics and multi-scale modular structure in networks," *arXiv preprint arXiv:0812.1770*, 2008.

[65] C. C. M. Grinstead and J. L. Snell, *Introduction to probability*. American Mathematical Soc., 1997.

[66] W. Pirie, "Spearman rank correlation coefficient," *Encyclopedia of statistical sciences*, 1988.

[67] T. Ho, M. Médard, J. Shi, M. Effros, and D. R. Karger, "On randomized network coding," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 41, no. 1. The University; 1998, 2003, pp. 11–20.

[68] C. Fragouli, J.-Y. Le Boudec, and J. Widmer, "Network coding: an instant primer," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 63–68, 2006.

[69] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," 2003.

[70] H. Trottier and P. Philippe, "Deterministic modeling of infectious diseases: theory and methods," *The internet Journal of infectious diseases*, vol. 1, no. 2, p. 3, 2001.

[71] M. Senftleben, M. Bucicoiu, E. Tews, F. Armknecht, S. Katzenbeisser, and A.-R. Sadeghi, "Mop-2-mop–mobile private microblogging," in *Financial Cryptography and Data Security*. Springer, 2014, pp. 384–396.

[72] W. X. Zhao, J. Jiang, J. Weng, J. He, E.-P. Lim, H. Yan, and X. Li, "Comparing twitter and traditional media using topic models," in *Advances in Information Retrieval*. Springer, 2011, pp. 338–349.

[73] G. K. Zipf, "The psycho-biology of language." 1935.