EFFICIENT MESSAGE DISSEMINATION FRAMEWORK FOR

DIVERSE WIRELESS NETWORKS

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Osman Khalid

In Partial Fulfillment
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Electrical and Computer Engineering

April 2014

Fargo, North Dakota

North Dakota State University
Graduate School

Title

EFFICIENT MESSAGE DISSEMINATION FRAMEWORK FOR DIVERSE WIRELESS NETWORKS

By

OSMAN KHALID

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Samee U. Khan
Chair

Dr. Ying Huang

Dr. Sudarshan Srinivasan

Dr. Jacob Glower

Approved:

June 4, 2014
Date

Dr. Scott C. Smith
Department Chair

ABSTRACT

Wireless networks exhibit diversity, ranging from mostly disconnected delay tolerant networks and partially connected mobile ad hoc networks, to mostly connected cellular networks. Besides having useful applications, including, vehicular communications, emergency response networks, battlefield networks, and wildlife monitoring, wireless networks face numerous challenges, such as unreliable connectivity, bandwidth restrictions, interference, frequent disruptions and delays, power outages, message loss, and malicious attacks. Moreover, when nodes are mobile, communication may be disrupted frequently for longer time periods. Designing protocols to tolerate such disruptions is challenging because of the extreme uncertainty in mobile wireless environments. Most of the existing approaches either require exact knowledge about future connectivity schedules, or perform message flooding in an attempt to improve message delivery rate. However, message flooding results in an increased overhead and loss of messages in resource constrained environments. Moreover, it is almost impossible to acquire precise future contact schedules in real-life scenarios.

The goal of this dissertation is to architect robust protocols that overcome disruptions and enable applications in diverse wireless networks. We propose a suite of protocols for wireless environments where nodes transfer messages during opportunistic contacts. To conserve resources, the protocols control flooding by autonomously adapting to the changing network conditions, to find optimal temporal routes between source and destination nodes. Moreover, the dissertation presents novel approaches that utilize time-series forecasting on nodes' contact patterns. Such routing schemes learn from nodes' temporal contacts and mobility patterns, and forecasts the future contact opportunities among the nodes. By making precise predictions about future contacts, messages are forwarded to only those nodes that increase the message delivery

likelihood. Simulation results proved that the proposed routing framework can be efficiently utilized in many real-life applications to disseminate delay tolerant data, such as electronic newspapers, weather forecasts, movie trailers, emergency information, and travel routes information in various parts of a city. The dissertation also proposes a novel application for mobile social networks that generates real-time recommendation of venues for a group of mobile users. The proposed framework utilizes Ant colony algorithm, social filtering, and hub and authority scores on the users' contextual information to produce optimal recommendations.

ACKNOWLEDGMENTS

I am grateful to acknowledge and thank all those who assisted me in my graduate program at North Dakota State University. I would like to express my deepest appreciation and a bundle of thanks to my academic advisor Dr. Samee U. Khan. His guidance, support, and patience throughout my years as a graduate student are truly appreciated. Special thanks to my other graduate committee members, Dr. Ying Huang, Dr. Sudarshan Srinivasan, and Dr. Jacob Glower. I would also like to express my gratitude to COMSATS Institute of Information Technology, Pakistan for their financial support.

Last but not the least, I would like to thank all my colleagues at NDSU for their kind help and support during my PhD.

DEDICATION

I would like to dedicate this dissertation to my parents, wife, brother, and sisters

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

Algorithm   Page

1. INTRODUCTION

1.1. Overview

The computer networks in today's era exhibit significant diversity in underlying communication mechanisms. A top level categorization of such mechanisms includes wired networks (such as local area networks) and wireless networks (such as cellular networks). Though wired networks have achieved significant advancement in technology, numerous challenges, such as long distance cabling costs, maintenance, and security issues limit the large-scale deployment of wired networks. For many years wireless networks have seen significant evolution in the communication technologies because of being economical, flexible, and ubiquitous in nature. Unlike wired networks, wireless networks do not require cables and can be easily deployed in an ad hoc manner enabling these networks to have a number of applications. A hierarchy and examples of wireless networks is presented in Fig.1.1.

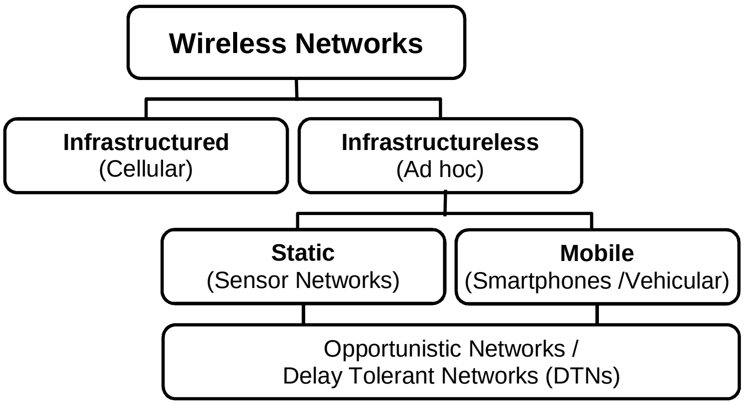


Fig. 1.1. Top level hierarchy of wireless networks.

Recent years have seen a prolific increase in network-enabled mobile devices. According to a market research report [1.1], in year 2009 alone a total of 144 million mobile phones were shipped with Wi-Fi capability and it is further estimated that such smart-phones may reach 66% of the total shipments till 2015 [1.2]. Multiple communication technologies in smart phones have enabled the users to easily communicate with or without the support of infrastructure [1.3]. In some environments, it may not be possible to deploy infrastructure-based networks (such as cellular or access points) because of the high deployment costs, security issues, and difficult terrains. Therefore, mobile devices may be required to communicate directly with each other without using the infrastructure, and together form on-the-fly networks, also known as Mobile Ad hoc NETworks (MANETs) [1.4] and Opportunistic Mobile Networks [1.5].

1.2. Motivation

The standard TCP/IP Internet is based on the assumption that a continuous bidirectional connection is available between source and destination to support end-to-end communication [1.6], [1.7]. The nodes in the Internet are connected most of the time, so the Internet protocols can easily estimate the links' costs between any two nodes, and packets are routed on least cost path. In case, the least cost path is disrupted, the alternate path is selected for message routing. However, designing protocols that can overcome disruptions in mobile environments is challenging because of the following reasons: **(a)** uncertain network conditions and **(b)** diversity in networks [1.3], [1.8].

Uncertainty in network conditions occurs mainly due to mobility of nodes, fluctuating channel conditions, short battery life, and bandwidth restrictions that may result in frequent topology changes/disconnections, unreliable connectivity, and transmission delays. Therefore, protocols for mobile environments need to make routing decisions based on partial knowledge about the network.

Mobile users require network access in diverse environments that range from fully disconnected to partially connected, and fully connected networks. Interconnecting these diverse networks poses several challenges due to heterogeneity of nodes and communication technologies. Designers need to first uncover the challenges specific to the network, and then design protocols that can address the challenges.

Due to the above mentioned issues, the traditional routing protocols (such as TCP/IP) designed for Internet are inapplicable for mobile environments, as end-to-end communication paths cannot be maintained among nodes. Although there exist a few routing protocols, such as AODV and DSDV [1.6] that create routing paths in dynamic environments, such protocols fail to work when the network is sparse and have frequent disconnections due to nodes' mobility. To improve the routing performance, several proposals, such as [1.9]-[1.17] have been presented that mainly addressed the routing/forwarding mechanisms for frequently disrupting mobile wireless networks. However, there is no consensus on which approach best suits a given scenario or application [1.3]. A few routing schemes perform message flooding in the network [1.11], [1.14], [1.15], [1.17]. Flooding increases the message delivery probability, but at the expense of network resources. On the contrary, the routing schemes that minimize the flooding, such as [1.10], [1.12], [1.13], [1.16], consume less resources, but at the expense of longer delays and decrease in message delivery probability.

In the light of above discussion, there is still a pressing need to develop resource conserving routing solutions for the mobile environments that must exhibit better message delivery rates with reduced network overhead. This dissertation addresses the critical area of resource efficiency in mobile routing, and exploits the nodes' mobility patterns to control message replicas within the network.

1.3. Contributions

The objective of our research is to architect robust and resource efficient protocols that can overcome disruptions, and enable applications in diverse communication networks.

Specifically, we address the issues and challenges pertaining to seamless message delivery in mobile networks that are prone to intermittent connectivity. We design a suite of protocols that allow the mobile users to opportunistically interconnect with or without the support of infrastructure to enable various real-life applications, such as dissemination of electronic newspapers, weather forecasts, movie trailers, emergency information, and travel routes information in various parts of a city with the help of mobile nodes. Mobility of nodes results in time evolving topology and frequent variation in network conditions. Therefore, the proposed routing protocols are designed to easily adapt to the changing network conditions, making the protocols resilient to network uncertainties.

Several of the previous studies reveal that humans follow repetitive schedule of meetings at similar places and times, and that the human mobility is predictable and follows a power-law distribution [1.18]. The aforementioned fact is further endorsed by Song *et al.* that the human mobility is 93% predictable [1.19]. To obtain benefits of repetitive mobility, we utilize time-series forecasting on nodes' contact patterns [1.20]. Therefore, our proposed protocols learn from the nodes' temporal contacts and mobility patterns, and forecast the future contact opportunities among the nodes. In this way, the protocols also control message flooding by forwarding the messages to only those nodes that increase the message delivery likelihood.

1.4. Network Types Considered

The types of networks considered in this dissertation are: **(a)** mostly disconnected, **(b)** intermittently connected, and **(c)** mostly connected [1.3], [1.8]. The networks are categorized on the basis of time durations of nodes' connections and disconnection in a particular network. Moreover, we define a network infrastructure as to be a cellular tower and/or access point (AP).

Most disconnected networks are also known as Delay Tolerant Networks (DTNs) [1.7], [1.21]. In such networks, infrastructure is difficult to deploy or is sparsely available, and nodes have infrequent connectivity. Nodes make contact for brief durations, for an interval of a few seconds to a few minutes, and stay disconnected for longer time durations for about hours or even days. The DTNs were initially designed to enable inter-planetary communication. Nowadays, such networks are also deployed in numerous terrestrial applications, such as wild life tracking, military networks, disaster relief networks, and vehicular ad hoc networks. In this dissertation, at first we present an empirical benchmarking of ten popular routing protocols for DTNs. As a next step, we propose three new protocols for DTNs based on the enhancements in the existing routing schemes. Our simulation results showed improved performance of the proposed techniques.

Intermittently connected networks have smaller durations of disconnections as compared to DTNs [1.3]. For instance, a mobile node making a connection with an AP installed at a bus station, the node will be disconnected when it moves out of range of the AP, and may be connected again after a while with an AP installed at a coffee shop. Though the disconnection durations are shorter than DTNs, the network is still not well connected. As a contribution in this dissertation, we proposed three routing schemes for intermittently connected networks. We utilize time-series forecasting on nodes' contact pattern to predict future contacts of nodes. Our results with real connection traces [1.22] as well as synthetic mobility indicated the better performance of the proposed schemes.

Mostly connected networks are assumed to have contemporaneous end-to-end communication paths among source and destination nodes. An example of such networks is the cellular networks and WiMAX, where mobile users stay connected to at least one cellular tower in range. One of the popular application areas of fully connected networks is mobile social networks [1.23]. The mobile social networks combine concepts from two separate disciplines: social network and mobile networks. The social network defines the structure of relationships and ties among mobile users. A novel application area of mobile social networks is the *recommender systems*. Such systems track the user activities, mobility patterns, and utilize the user's contextual information to provide recommendations on a variety of items. In this dissertation, we build a novel cloud-based framework to perform real-time recommendations of venues to the mobile subscribers. The proposed framework recommends new venues to a user based on the contextual information and similarities of interest with the friends in the user's social network.

1.5. List of Publications

Some of the contributions presented in this dissertation have appeared in the following publications:

1.5.1. Published

1. **O. Khalid**, M. U. S. Khan, S. U. Khan, A. Y. Zomaya, "OmniSuggest: A Ubiquitous Cloud based Context Aware Recommendation System for Mobile Social Networks", *IEEE Transactions on Services Computing*, 10.1109/TSC.2013.53, December, 2013.
2. **O. Khalid,** S. U. Khan, S. A. Madani, K. Hayat, M. I. Khan, N. Min-Allah, J. Kolodziej, L. Wang, S. Zeadally, and D. Chen, "Comparative Study of Trust and Reputation Systems for Wireless Sensor Networks," *Security and Communication Networks*, vol. 6, no. 6, pp. 669-688, 2013.
3. K. Bilal, S. U. R. Malik, **O. Khalid**, A. Hameed, E. Alvarez, V. Wijaysekara, R. Irfan, S. Shrestha, D. Dwivedy, M. Ali, U. S. Khan, A. Abbas, N. Jalil, and S. U. Khan, "A Taxonomy and Survey on Green Data Center Networks," *Future Generation Computer Systems*, http://dx.doi.org/10.1016/j.future.2013.07.006
4. **O. Khalid,** S. U. Khan, J. Kolodziej, L. Zhang, J. Li, K. Hayat, S. A. Madani, L. Wang, and D. Chen, "A Checkpoint Based Message Forwarding Approach for Opportunistic

Communication," in *26th European Conference on Modeling and Simulation (ECMS)*, Koblenz, Germany, May 2012, pp. 512-518.

5. **O. Khalid**, S. U. Khan, S. A. Madani, K. Hayat, L. Wang, D. Chan, and R. Ranjan, "Opportunistic Databank: A Context-aware on-the-fly Data Center for Mobile Networks," in Handbook on Data Centers, S. U. Khan and A. Y. Zomaya, Eds., Springer-Verlag, New York, USA. (Forthcoming.)

6. **O. Khalid**, K. Bilal, and S. U. Khan, "Green Computing," *IEEE Technical Committee on Scalable Computing Blog*, April 16, 2012

1.5.2. Submitted

1. **O. Khalid**, M. Sualeh, R. N. B. Rais, K. Hayat, S. A. Madani, J. Kolodziej, F. Zhang, R. Ranjan, N. Ghani, D. Chen, L. Wang, S. U. Khan, and A. Y. Zomaya, "Benchmarking and Modeling of Routing Protocols for Delay Tolerant Networks," Submitted to *future generation of computer systems*, January, 2014.

2. **O. Khalid**, R. N. B. Rais, N. Ghani, and S. U. Khan, "Forecast and Relay: A Message Routing Scheme for Opportunistic Mobile Networks," Submitted to *IEEE communication letters*, March, 2014.

3. **O. Khalid**, N. Ghani, and S. U. Khan, "APS: An Adaptive Prognostic Message Routing Scheme for Delay Tolerant Networks," Submitted to *IEEE communication letters,* March, 2014.

4. S. Khan, **O. Khalid**, M. I. Khan, S. U. Khan, M. Azeem, M. N. Saqib, and S. Tanveer, "Trust-based Secure and Optimized Link State Routing Protocol," Submitted to *IET Information Security*, 2013.

5. R. Irfan, **O. Khalid**, M. U. S. Khan, C. Chira, R. Ranjan, F. Zhang, S. U. Khan, B. Veeravalli, K. Li, and A. Y. Zomaya, “A Context-aware Recommendation Framework based on Bi-objective Optimization Techniques for Mobile Social Networks,” Submitted to *IEEE transactions on services computing*, 2014.
6. K. Bilal, **O. Khalid**, S. U. R. Malik, M. U. S. Khan, S. U. Khan, and A. Zomaya, “Fault Tolerance in the Cloud,” Submitted to *Encyclopedia on Cloud Computing,* Wiley 2014

1.6. Dissertation Outline

The dissertation is organized as follows. In Chapter 2, we present the background and related literature. Chapter 3 presents an empirical benchmarking of ten DTN routing protocols. Moreover, we propose three new routing schemes for DTNs in Chapter 3. A routing scheme based on time-series forecasting is proposed for intermittently connected opportunistic mobile networks in Chapter 4. In Chapter 5, we propose a checkpoint-based message routing approach for DTNs. Chapter 6 presents a data replication scheme for mobile networks. A routing protocol based on time-series forecasting for DTNs is proposed in Chapter 7. In Chapter 8, we present a venue recommendation system for mobile social networks, and conclusions with future research directions are presented in Chapter 9.

1.7. References

[1.1] X A. Ionut, “Wi-fi enabled mobile phone handsets in the US, 2010-2015,” *Technical report. Market research analysis*, Coda Research Consultancy, 2010.

[1.2] A. Ionut, “Wi-fi capable handsets: Residential and enterprise markets for wi-fi and dual-mode handsets,” *Technical report,* ABI Research, 2009.

[1.3] M. Conti and M. Kumar, “Opportunities in Opportunistic Computing,” *IEEE Computer,* vol. 43, no. 1, pp. 42-50, 2010.

[1.4] S. U. Khan, "Mosaic-Net: A Game Theoretical Method for Selection and Allocation of Replicas in Ad Hoc Networks," *Journal of Supercomputing*, vol. 55, no. 3, pp. 321-366, 2011.

[1.5] J. Whitbeck, Y. Lopeza, J. Leguaya, V. Conana, and M. D. Amorim, "Push-and-track: Saving infrastructure bandwidth through opportunistic forwarding," *Pervasive and Mobile Computing*, vol. 8, no. 5, pp. 682–697, 2012.

[1.6] C. P. Mayer, "Hybrid Routing in Delay Tolerant Networks," *KIT Scientific Publishing*, 2012.

[1.7] Delay-Tolerant Networking Research Group (DTNRG), http://www.dtnrg.org, accessed March, 2014.

[1.8] A. Balasubramanian, "Architecting protocols to enable mobile applications in diverse wireless networks," Ph.D. dissertation, Dept. Comp. Sci., University of Massachusetts Amherst, 2011.

[1.9] S. M. Bilal, A. R. Khan, S. U. Khan, S. A. Madani, B. Nazir, M. Othman, "Road Oriented Traffic Information System for Vehicular Ad hoc Networks," *Wireless Personal Communications*. DOI: 10.1007/s11277-014-1651-0.

[1.10] J. Kolodziej, S. U. Khan, L. Wang, N. Min-Allah, S. A. Madani, N. Ghani, and H. Li, "An Application of Markov Jump Process Model for Activity-Based Indoor Mobility Prediction in Wireless Networks," in 9th *IEEE International Conference on Frontiers of Information Technology (FIT)*, Islamabad, Pakistan, 2011, pp. 51-56.

[1.11] A. Balasubramanian, B. Levine, and A. Venkataramani, "Replication Routing in DTNs: A Resource Allocation Approach" *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 596-609, 2010.

[1.12] P. Cheng, K. C. Lee, M. Gerla, and J. Härri, "GeoDTN+Nav: Geographic DTN Routing with Navigator Prediction for Urban Vehicular Environments," *Mobile Networks and Applications*, vol. 15, no.1, pp. 61-82, 2010.

[1.13] P. Hui, J. Crowcroft, and E. Yoneki "BUBBLE Rap: Social-Based Forwarding in Delay Tolerant Networks," *IEEE Transactions on Mobile Computing*, vol.10, no.11, pp. 1576-1589, 2011.

[1.14] S. C. Nelson, M. Bakht, and R. Kravets, "Encounter-Based Routing in DTNs," *IEEE Infocom*, 2009, pp. 846-854.

[1.15] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "Probabilistic Routing Protocol for Intermittently Connected Networks," http://tools.ietf.org/html/rfc6693, 2012, accessed on Feb 2014.

[1.16] Z. Guo, B. Wang, and J. Cui, "Generic prediction assisted single-copy routing in underwater delay tolerant sensor networks," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1136–1149, 2013.

[1.17] J. Ott, A. Keränen, and E. Hyytiä, "BeachNet: Propagation-based Info-Sharing in Mostly Static Networks," In *Proceedings of ExtremeCom.*, 2011.

[1.18] 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*, vol. 19, no. 3, pp. 630-643, 2011.

[1.19] C. Song, Z. Qu, N. Blumm, and A. Barabási, "Limits of Predictability in Human Mobility," *Science Magazine*, vol. 327, no. 5968, pp. 1018-1021, 2010.

[1.20] Forecasting: principles and practice, http://otexts.com/, accessed on March 2014.

[1.21] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz, "Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges," *IEEE Communications Surveys and Tutorials,* vol. 14, no. 2, pp. 607-640, 2012.

[1.22] Infocom06 connectivity traces on "CRAWDAD website," http://crawdad.cs.dartmouth.edu, accessed on 05 March, 2014.

[1.23] N. Vastardis and K. Yang, "Mobile Social Networks: Architectures, Social Properties, and Key Research Challenges," *IEEE communications surveys and tutorials,* vol. 15, no. 3, pp. 1355-1371, 2013.

2. BACKGROUND AND RELATED WORK

This chapter presents the background as well as the literature survey on recent works related to the topics investigated throughout this dissertation. We perform categorization of the existing message dissemination schemes proposed for various types of mobile networks.

2.1. Background

Wireless radio range variations, limited energy resources, sparsity of mobile nodes, continuous mobility, and noise, to name a few, are the reasons due to which mobile networks suffer from frequent disconnections. This phenomenon is undesirable when mobile hosts are accessing data from each other. As it is not possible to control randomly occurring network disconnections, an alternative solution to this problem is to replicate multiple copies of data onto various mobile hosts so that when disconnections/disruptions occur, mobile hosts can still access data [2.1], [2.2]. Replication process distributes additional copies of primary data items into the network in order to increase accessibility and decrease communication costs. In the past few years, data replication has been studied extensively for both the MANETs and DTNs environments [2.1], [2.3], [2.4]–[2.10]. Fig. 2.1(a) and Fig. 2.1(b) illustrate the replication scenario in MANETs.

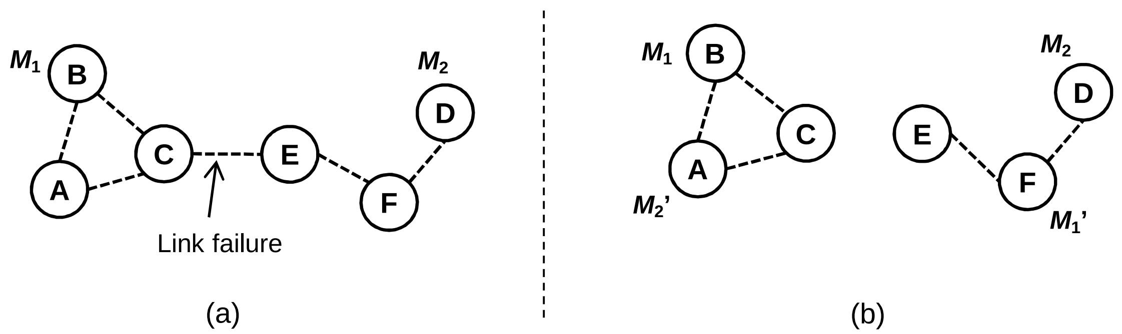


Fig. 2.1. Replication example in MANETs: (a) network division and data access, and (b) effective data replication for continued data access.

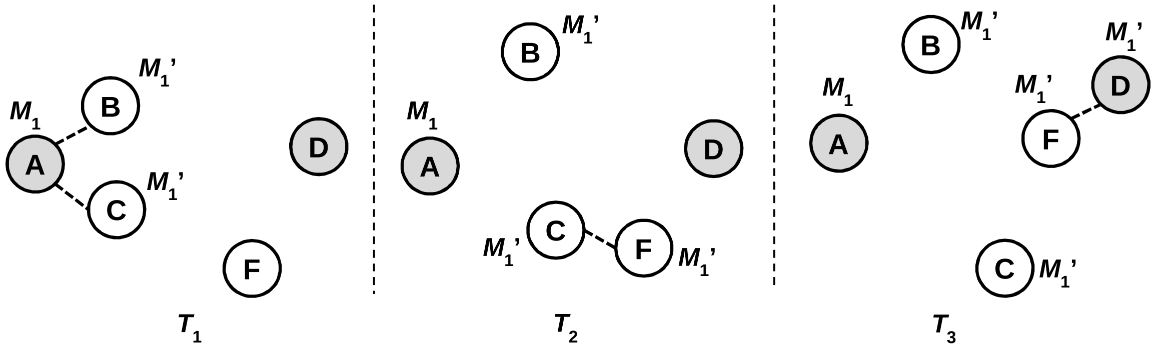


Fig. 2.2. Replication example in DTNs. Data is replicated from node to node during opportunistic contacts at time slots T1, T2, and T3, without any global knowledge of network topology.

If the central link between nodes *C* and *E* fails, then the set of mobile hosts *E*, *F*, and *D* will not be able to access the data item M_1. Similarly, the data item M_2 will be inaccessible by the nodes *A*, *B*, and *C*. To cope with the problem of data inaccessibility due to network division, one possible solution is to create replicas of original messages M_1 and $M_{2,}$ and place these replicas at the opposite sides of the ad hoc network. In this way, every mobile host can access both data items even after the network division, as indicated in Fig. 2.1(b). Due to the existence of end-to-end communication paths, the aforementioned mechanism of proactively placing replicas on nodes before the link failures is possible only in the case of MANETs. However, replica placement exhibits more complexity when the network is sparse with no end-to-end communication paths among nodes, as in the case of DTNs.

Fig. 2.2 illustrates an example scenario of replica allocation in DTNs. For the sake of simplicity, we consider allocation of a single message M_1 from node *A* to node *D*. At a reference time T_1, node *A* is not in the communication range of node *D*. On making contact with nodes *B* and *C*, suppose the node *A* is not sure which of the two nodes *B* and *C* will make contact with node *D* in future. Therefore, node *A* places a replica of M_1 on both the mobile nodes *B* and *C*.

During time T_2, only the node C makes contact with another node, and replicates M_1 on node F. Finally, node F transfers replica on node D on making contact at time slot $T_{3.}$

It is evident from the given example that replica placement in DTNs is dependent on the occurrence of opportunistic contacts among mobile nodes. Therefore, the main difference in data replication between MANETs and DTNs is the absence of any centralized mechanism and/or global knowledge of network for DTNs. Moreover, in any of MANET/DTN network, the decision of where to place replica must trade off the cost of accessing data that is reduced by additional copies with the cost of storing and updating the replicas [2.1], [2.4]. These costs have severe implications in ad hoc network environments because mobile hosts have limited resources (energy, storage, and processing power). Therefore, efficient and effective replication schemes strongly depend on how many replicas to be placed in the system, and to what nodes [2.1].

2.2. Message Dissemination Schemes for MANETs

Khan *et al.* in their celebrated work have rigorously addressed message dissemination problem in distributed systems [2.1]–[2.3], [2.11], [2.12]. Specifically, in [2.1], [2.12] Khan *et al* pioneered in applying game theory to ad hoc network replica allocation problem (ADRP). In [2.1], the authors proposed a novel scheme that seeks to strategically balance energy, bandwidth, and storage space through a cooperative game-theory approach for replication in a mobile environment. In the presented work [2.1], the authors: **(a)** derived a mathematical problem formulation for ADRP, **(b)** proposed an optimization technique that allocates replicas so as to minimize the network traffic under storage constraints with “read from the nearest” and “push based update through the primary mobile server” policies, and **(c)** used a strict consistency model as opposed to an opportunistic consistency model. The authors addressed selfish behavior of mobile servers in the proposed solution. In ad hoc networks, resources may belong to different self-interested servers. These servers may manipulate the resource (replica) allocation mechanism for their own benefit by misrepresenting their preferences, which may result in severe performance degradation. The proposed technique involved players (mobile hosts) that compete through bids in a non-cooperative environment to replicate data objects that are beneficial to themselves and the system as a whole. It is always possible that in order to satisfy local queries, the players replicate data objects that are not beneficial to the system as a whole in terms of saving communication cost (although it may be productive from the players' point of view). To counter such negative notions, a referring body was introduced (termed as the mechanism). The aim of the mechanism was to direct the competition in such a fashion that a global optimal was achieved even though the agents are competing against one another. Moreover, the basic objective of the proposed work was to make the system robust against incorrect dissemination of information by the players. To cater for the possibility of collusive behavior of the players, the scheme used the *Vickrey* payment protocol that leaves the players with no option other than to bid in such a fashion that is beneficial to the system as a whole. The goal of a player is to maximize its profit, which is payment minus cost. The goal of the mechanism is to minimize the total data item transfer cost in the network due to the read and update accesses. In Mosaic-Net [2.1], the authors used side payments to encourage players to tell the truth. The authors [2.1] proved that the ADRP problem in general is NP-complete and also identified some useful properties of the proposed scheme and the necessary conditions of optimality.

Hirsch *et al.* [2.4] proposed a game-theory based model for ADRP, where all nodes were assumed to be cooperative. The authors applied ideas from *Volunteer's dilemma* [2.13] in area of game theory. Under the Volunteer's dilemma approach, a node volunteers to store replicas that will incur some cost to the node in terms of its resources, but in return will benefit the resource conservation and lifespan of the whole network. The proposed approach performed volunteer nodes' selection for replica assignment in such a manner that a global utility function defining the network cost is optimized. In the proposed algorithm, named as Cooperative Altruistic Data Replication (CADR), the net global benefit (NGB) is calculated for each node on the return path of requested replica, where NGB depends on two parameters: **(a)** global savings (GS) and **(b)** global cost (GC). The GS is the global network savings when the node makes a local replica of data item to minimize traffic through read requests. The GC is the cost incurred when data item is updated, or displaced from primary node to other node when primary node is low on resources. The CADR algorithm proceeds as illustrated in the following. On return path of a requested replica k, each node i calculates NGB as $NGB_i^k = GS_i^k - GC_i^k$, and stores NGB_i^k into a matrix appended in the header of response replica. When replica is received by the requesting node r, the node computes NGB_r^k. Then, if $NGB_r^k > NGB_i^k, \forall i$, then the node r stores a copy of data item in its buffer. Otherwise, replica is placed on a node i on the request/response path, such that $NGB_i^k > NGB_j^k, \forall j$.

It is important to observe from the above described techniques addressing ADRP that most of the approaches utilize a common assumption of availability of global network knowledge. Such global network knowledge constitutes the following information: **(a)** number of replicas of original data items, **(b)** the identifiers of nodes having original and copies of data items, **(c)** frequency of access of each replica, and (d) frequency of contacts among various nodes. However, it is formally proved by Khan *et al.* [2.1] that despite the availability of global information, the varying dynamics of network topology in MANETs make replica allocation problem NP hard. The things get further complicated in DTNs, due to lack of global network state information, as well as scarcity of end-to-end communication paths. In the following section we address the replica allocation challenges in DTNs and propose our solution to the problem.

2.3. Message Dissemination Schemes for DTNs

DTNs are resource-constrained networks in terms of transfer bandwidth, energy, and storage. Recently, several works, such as [2.5], [2.8], [2.9], [2.14], [2.15] have been presented on the routing/forwarding mechanisms for the OMNs. However, there is no consensus on which approach best suits a given scenario or application. Among all the routing strategies, the single-copy forwarding schemes are considered to be the most resource conservative, as such approaches are designed to forward only a single copy of the message within the network [2.15]. However, such approaches suffer from delay and reduced message delivery, as single message copy may have to wait longer. In some cases, the message may never reach the destination. The flooding-based approaches spread multiple replicas of a message within the network [2.5], [2.8], [2.10]. A higher number of message replicas improves the message delivery probability, but at an increased expense of network resources. The authors in [2.16] set a limit on the number of replicas per message that lowered the overhead, but increased the message delay. Lindgren *et al.* [2.9] proposed a probabilistic message replication approach, named as the *PRoPHET*, where a node replicates a message to a neighbouring node, if and only if, the neighbouring node has more frequently encountered with the destination. However, the *PRoPHET* protocol is not mobility-cognizant and sets no limit on the number of message replicas. To address network overhead, the techniques proposed in [2.8] and [2.14] restrict a message replica from entering a node's buffer for a certain period of time. However, such a restriction results in increased delay as messages are restrained from quickly spreading within the network.

For many years human mobility has been an active area of research in DTNs. It has been shown through various experiments that human populations follow repeated mobility patterns.

Different methods have been applied to collect the human mobility traces. For example Rhee *et al*. [2.17] studied the urban human mobility through GPS traces. Cacciapuoti *et al*. [2.18] used the signaling information in cell phones through the AirSage (www.airsage.com) technology to gather the mobility traces. All the previous studies reveal that humans tend to follow a repetitive schedule of meetings at same places and times, and the human mobility is predictable following a power law distribution. The aforementioned fact is further endorsed by Song *et al*. [2.19] that human mobility is 93% predictable. Therefore, CP based approach presented in Chapter 5 is inspired by the same fact that humans tend to follow schedule of meetings with higher interaction probabilities at commonly visited places such as bus stops/stations and shopping malls where the CPs may be deployed.

2.4. Mobile Social Networks

In this section, we discuss some of the recently proposed (2009-2013) techniques for venue recommendation systems in mobile social networks. The existing approaches can be categorized as [2.20]: **(a)** trajectory based, **(b)** explicit rating based, and **(c)** check-in based approaches. Trajectory based approaches utilize information about a user's visit sequence to various locations, the paths selected, and the duration of stays. Doytsher *et al.* [2.21] proposed a trajectory-based graphical model that keeps track of frequently traveled routes by users and recommend best route to a new user. The authors in [2.22] mine GPS trajectories data to extract most popular locations based on users' travel sequences. Although the aforementioned approaches suggests locations based on users' past trajectories, they are unable to distinguish the places in terms of their categories, which we performed in our proposed *OmniSuggest* framework in Chapter 8. Moreover, such approaches suffer from data sparseness problems, as usually a person does not frequently visit on multiple places.

Many online social services, such as Yelp (yelp.com) and Yellow pages (yellowpages.com) allow users to rate the visited locations. Rating-based recommendation systems utilize the existing ratings' data to recommend people with most popular venues or travel routes in a city. The authors in [2.23] proposed models based on collaborative filtering that take into account users' existing ratings to generate personalized venue recommendations. The aforementioned approaches may closely capture users' preferences, but are not scalable enough to simultaneously process huge volumes of real-time data. Moreover, they also suffer from data sparseness issues due to limited number of entries within the user-rating matrix.

Most of the above mentioned approaches have designs built on (memory based) CF models, which enables these approaches to depict a user's future preferences based on his/her past entries. However, these approaches suffer from scalability issues due to large number of similarity computations on user-venue matrix during online recommendation process. Moreover, such approaches also suffer from data sparseness and cold start problems, as there are very few users who have visited large number of venues. Furthermore, these approaches do not provide a solution to the group recommendation problem as well as do not take into account the effect of real-word time-varying conditions on recommendations. To address these limitations, our proposed cloud based recommendation framework, *OmniSuggest*, presents a solution for scalability, data sparseness, and group recommendation challenges. The proposed approach also takes into account the real-world conditions while generating recommendations that results in a set of venues that satisfies all of the group members.

2.5. References

[2.1] S. U. Khan, "Mosaic-Net: A Game Theoretical Method for Selection and Allocation of Replicas in Ad Hoc Networks," *Journal of Supercomputing*, vol. 55, no. 3, pp. 321-366, 2011.

[2.2] S. U. Khan and I. Ahmad, “A Pure Nash Equilibrium based Game Theoretical Method for Data Replication across Multiple Servers,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 4, pp. 537-553, 2009.

[2.3] S. U. Khan and C. Ardil, “A Competitive Replica Placement Methodology for Ad Hoc Networks,” in *International Conference on Parallel and Distributed Computing Systems (ICPDCS)*, Oslo, Norway, July 2009, pp. 128-133.

[2.4] D. Hirsch and S. Madria, “Data Replication in Cooperative Mobile Ad-Hoc Networks,” *Mobile Networks and Applications*, vol. 18, no. 2, pp. 237-252, 2013.

[2.5] A. Balasubramanian, B. Levine, and A. Venkataramani, “Replication Routing in DTNs: A Resource Allocation Approach,” *IEEE/ACM Transactions on Networking*, vol.18, no.2, pp.596-609, 2010.

[2.6] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz, “Disruption-tolerant networking: A comprehensive survey on recent development and persisting challenges,” *IEEE Communications Surveys Tutorials*, vol. 14, no. 2, pp. 607–640, Second Quarter, 2012.

[2.7] O. Khalid, S. U. Khan, J. Kolodziej, L. Zhang, J. Li, K. Hayat, S. A. Madani, L. Wang, and D. Chen, “A checkpoint based message forwarding approach for opportunistic communication,” *European Conference of Modeling and Simulation*, 2012.

[2.8] J. Ott, A. Keränen, and E. Hyytiä, “BeachNet: Propagation-based Information Sharing in Mostly Static Networks,” in *Proc. of ACM ExtremeCom*, 2011.

[2.9] A. Lindgren, A. Doria, A. Davies, and S. Grasic, “Probabilistic Routing Protocol for Intermittently Connected Networks,” http://tools.ietf.org/html/rfc6693, 2012, accessed on March 2014.

[2.10] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *Technical Report CS-200006, Duke University*.

[2.11] S. U. Khan, "On a Game Theoretical Methodology for Data Replication in Ad Hoc Networks," in *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, Las Vegas, NV, USA, July 2009, pp. 232-238.

[2.12] S. U. Khan, A. A. Maciejewski, H. J. Siegel, and I. Ahmad, "A Game Theoretical Data Replication Technique for Mobile Ad Hoc Networks," in *22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Miami, FL, USA, April 2008.

[2.13] S. Lee, D. Levin, V. Gopalakrishnan, B. Bhattacharjee, "Backbone construction in selfish wireless networks." In *ACM SIGMETRICS*, New York, 2007, pp. 121–132.

[2.14] C. P. Mayer, "Hybrid Routing in Delay Tolerant Networks," *KIT Scientific Publishing*, 2012.

[2.15] Z. Guo, B. Wang, and J. Cui, "Generic prediction assisted single-copy routing in underwater delay tolerant sensor networks," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1136–1149, 2013.

[2.16] T. Spyropoulos, T. Turletti, and K. Obraczka, "Routing in Delay-Tolerant Networks Comprising Heterogeneous Node Populations," *IEEE transactions on mobile computing*, vol. 8, no. 8, pp. 1132-1147, 2009.

[2.17] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong, "On the Levy-Walk Nature of Human Mobility Networking," *IEEE/ACM Trans on Networking*, vol. 19, no. 3, pp. 630-643, 2011.

[2.18] A. S. Cacciapuoti, F. Calabrese, M. Caleffi, G. Di Lorenzo, and L. Paura, "Human mobility-enabled wireless networks for emergency communications during special events," *Pervasive and Mobile Computing*, vol. 9, no. 4, pp. 472–483, 2013.

[2.19] C. Song, Z. Qu, N. Blumm, and A. Barabási, "Limits of Predictability in Human Mobility," *Sci. Mag,* vol. 327, no. 5968, pp.1018-1021.

[2.20] J. Bao, Y. Zheng, M.F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," In *Proc. 20th International Conference on Advances in Geographic Information Systems*, ACM New York, pp.199-208, 2012.

[2.21] Y. Doytsher, B. Galon, and Y. Kanza, "Storing routes in socio-spatial networks and supporting social-based route recommendation," In *Proc. 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, ACM, pp. 49-56, 2011.

[2.22] Y. Zheng, L. Zhang, X. Xie, and W.Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," In *Proceedings of the 18th international conference on World wide web*, ACM, pp. 791-800, 2009.

[2.23] L. Wei, Y. Zheng, and W. Peng, "Constructing popular routes from uncertain trajectories," In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 195-203, 2012.

3. BENCHMARKING AND MODELING OF ROUTING PROTOCOLS FOR DELAY TOLERANT NETWORKS

This paper is submitted to the journal: *Future Generation Computer Systems* (FGCS). The authors of this paper are Osman Khalid, Muhammad Sualeh, Rao Naveed Bin Rais, Khizar Hayat, Sajjad A. Madani, Joanna Kolodziej, Fan Zhang, Rajiv Ranjan, Nasir Ghani, Dan Chen, Lizhe Wang, Samee U. Khan, and Albert Y. Zomaya.

3.1. Overview

Delay Tolerant Networks (DTNs) are characterized by their intermittent connectivity, frequent partitioning, long message delays, and lack of end-to-end communication paths. In the year 2002, the Internet Research Task Force (IRTF) documented a DTN architecture specification for interplanetary communication [3.1], [3.2]. Later on, in addition to the space communication, the focus of DTN research also shifted towards establishing communications in the challenging terrestrial network environments, such as sparsely distributed mobile ad-hoc networks and wireless sensor networks. Fig. 3.1 presents a few application areas of DTNs.

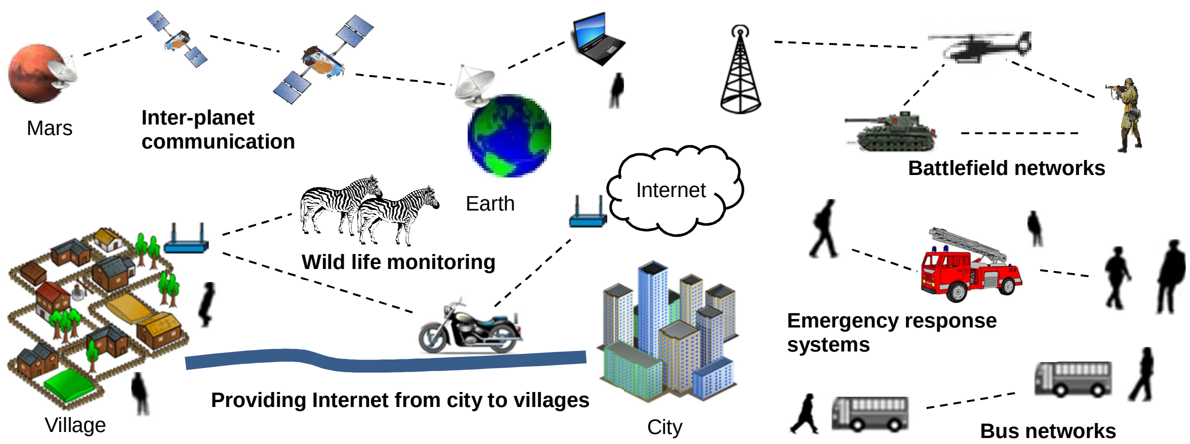


Fig. 3.1. Examples of DTNs where devices have intermittent connectivity.

3.1.1. Motivation

Wireless radio range variations, limited energy resources, sparsity of mobile nodes, and noise, to name a few, are the reasons due to which such networks suffer from frequent disruption and delay in the process of message transfer. The traditional mobile ad hoc network (MANET) routing protocols, such as OLSR [3.24], AODV [3.25] and DSR [3.26] are not suitable for network environments with frequent disruptions, as such protocols require the existence of end-to-end communication paths among sources and destination nodes. The inherent uncertainty about network conditions makes message routing in DTNs a daunting task. In the absence of end-to-end routing paths, nodes have to rely on opportunistic contacts to exchange messages. There is no guarantee that a message eventually reaches the destination, as the message may be dropped due to network congestion, or life time expiry, yielding a best effort delivery service. Therefore, it is quite difficult for a DTN routing protocol to achieve a 100% message delivery.

The typical performance metrics for DTN routing are: **(a)** message delivery ratio, **(b)** message latency, and **(c)** message overhead. Delivery ratio is the percentage of messages delivered successfully. A message's latency is the total time spent between message creation and delivery to the destination. Overhead indicates the number of extra transmissions for each delivered message. Primary objective of a DTN protocol is to improve message delivery ratio with minimum latency and overhead, in a network with limited resources. Therefore, DTN protocols adopt various strategies and heuristics to improve the routing performance, and are generally categorized as [3.3]: **(a)** deterministic, **(b)** enforced, and **(c)** opportunistic. Among the aforementioned categories, opportunistic routing is the most challenging, as it is applied to networks where nodes' mobility information is not known beforehand, and nodes have to rely on random contacts for message transfer. Moreover, to transfer data to potential intermediate relay nodes, opportunistic routing utilizes various heuristics to find the suitability and fitness of a relay, such as **(a)** relay affiliation with community, **(b)** contact duration, **(c)** available bandwidth, **(d)** available storage at relay nodes, **(e)** message expiration time, and **(f)** message priority [3.3], [3.4]. An ever increasing number of protocols addressing "opportunistic" DTN scenarios have been proposed in the past. However, still it is not much clear how existing solutions can be applied to a variety of DTN applications, given their requirements and underlying network characteristics. Moreover, to this date, there is no specific study that performs large scale comparison of different routing schemes and gives a solid argument against or in favor of a specific protocol in a particular scenario and underlying network characteristics.

3.1.2. Contributions

In recent years, numerous comparative studies have been conducted for DTN protocols, under various parameters, simulation platforms, and mobility scenarios [3.3], [3.5]–[3.7]. All such studies to some extent presented useful comparisons for DTN protocols. However, most of the evaluations were limited in terms of: **(a)** number of protocols compared, **(b)** simulation parameters, and **(c)** DTN scenarios. We address this very problem in this chapter by presenting a thorough empirical comparison of ten carefully chosen "opportunistic" routing protocols. These ten protocols were selected because they seemed to be most appropriate in addressing the routing challenges for diverse DTN environments [3.8], [3.9]. Moreover, the protocols cover a broad range of routing scenarios, such as, single-copy forwarding, replication/flooding, probabilistic routing, greedy, and nature inspired content dissemination techniques. The protocols are evaluated with Opportunistic Network Environment simulator (ONE) [3.10] by using both real-world traces [3.11] and synthetic mobility scenarios. Moreover, evaluations are done for performance metrics, such as **(a)** delivery ratio, **(b)** latency, and **(c)** overhead, so as to fully understand the strengths and weaknesses of these routing techniques. To the best of our knowledge, this is the first study that presents an extensive benchmarking of DTN protocols under diverse network conditions and parameters.

In addition to the benchmarking, we go one step further by proposing three DTN routing protocols, namely, Enhanced Epidemic Scheme (EES), Adaptive Multi-Copy Spray (AMS), and Adaptive Source Token Multi-Copy Spray (ASTMS), by improving the models of the existing routing schemes. The presented techniques efficiently utilize the available network information to control message replication frequency, by autonomously adapting to the varying network conditions, and find optimal spatiotemporal routes for messages. The controlled distribution of message copies helps the efficient utilization of network resources, resulting in the improved delivery ratio and overhead performance. In summary, our contributions in this chapter are as follows:

(1) We present the basic building blocks of DTN routing and discuss the various critical components that must be considered in designing of an efficient DTN protocol.

(2) We perform the empirical benchmarking of ten carefully-selected DTN routing schemes to evaluate their performance under the similar platform and scenarios. This provides a good and useful comparative study of routing protocols in terms of their strengths and weaknesses.

(3) The routing protocols are evaluated by varying different parameters, such as: **(a)** number of nodes, **(b)** message creation rate, **(c)** buffer size, **(d)** message size, and **(e)** bandwidth. The evaluation results are generated by using both the synthetic mobility models and the real mobility traces.

(4) We propose three new DTN routing protocols by enhancing the models of the existing schemes. By introducing adaptability features in the proposed schemes, the protocols are able to control message replication frequency by autonomously adapting to the time

varying network conditions. Our results indicate significant improvement in performance of the proposed schemes in terms of delivery ratio and overhead.

The rest of the chapter is organized as follows. Section 3.2 demonstrates DTN routing problem and discusses the important factors that must be carefully considered in modeling of a DTN protocol. In Section 3.3, we present a comparative analysis of the ten selected DTN routing protocols with an emphasis on the strengths and weaknesses of each the protocol. The simulation and benchmarking of the protocols under real-world traces and synthetic mobility is performed in Section 3.4. In Section 3.5, we present the models of the three new routing schemes and analyze their performance in comparison with existing schemes.

3.2. DTN Routing Insights

The routing problem in DTNs can be expressed as: *"Which messages to transfer during an opportunistic contact, such that, they contribute to overall improvement of network performance parameters, such as communication overhead, delivery ratio, and delay?"* It is quite challenging to find a precise answer to this question as the routing performance is affected by many factors, such as: **(a)** message size, **(b)** message rate, **(c)** message life-time, **(d)** buffer size, **(e)** bandwidth, **(f)** transmission range, **(g)** interference, **(h)** node speed, **(i)** node energy, **(j)** mobility pattern, **(k)** node's sleep intervals, and **(l)** network size. The numerous combinations and values of the aforementioned factors constitute the different DTN scenarios. The applicability of a DTN protocol for various scenarios depends on the number of aforementioned factors considered while designing a protocol. In the following subsections, we define the basic building blocks of DTN routing and discuss various critical factors that affect the performance of a DTN protocol.

3.2.1. Forwarding versus Replication

Message transfer in DTN routing is achieved through either *forwarding* or *replication* [3.12]. When a message is *forwarded* to a neighbor, the sending node deletes the local copy of message from the buffer. This way, only a single copy of message stays in the network [3.12]. Alternatively, when the message is *replicated*, both the sending and receiving nodes carry a separate copy of the same message. Forwarding yields minimum overhead and consumes less buffer space as there are lesser number of messages in the network. However, the decrease in message replicas also decreases the probability that a message will be delivered to the destination. Therefore, forwarding is mostly applicable for *deterministic* DTN environments, where nodes' mobility patterns are known beforehand or can be precisely predicted. Message replication frequency may be *controlled* [3.13] or *uncontrolled* [3.14]. The controlled replication scheme replicates a message only when a certain condition is satisfied. For example, the neighbor node is more likely to encounter with the destination node as compared to the node carrying the message. The uncontrolled replication is a flooding based technique in which maximum copies of the same message are floated unconditionally in the network. Increase in message copies also increases probability of a message reaching the destination, but at the cost of higher overhead and buffer consumption. Therefore, increased replication rate may also increase the message drop due to buffer overflows, which may reduce the overall delivery ratio.

3.2.2. Metadata Exchange

The routing protocols differ in the way they utilize the amount of information or *metadata* to perform message transfer decisions [3.3]. In many protocols, nodes maintain a record of their contacts with other nodes in the form of a list $L\{i, t_c, t_d, b\}$ having parameters: **(a)** node identification i, **(b)** contact time t_c, contact duration t_d, and **(c)** bytes transferred b, respectively [3.6], [3.15]. Based on the metadata, a node computes optimal routes for each message. The *uncontrolled replication-based* routing schemes do not utilize metadata [3.12], [3.14] as compared to the *controlled replication-based* routing schemes [3.6], [3.15], which utilize metadata to improve performance at the cost of increased complexity and computational requirements. When two nodes make a contact, the nodes share and update their respective metadata. This enables the information symmetry throughout the network.

3.2.3. Buffer Management

Routing schemes also differ in the way they employ buffer management policies. The buffer management typically includes: **(a)** message queue sorting and **(b)** message deletion. Message queue sorting is performed by assigning priorities to the messages, and a message of least priority may be deleted. One factor that influences the buffer management is available *contact time window*, which is the time duration between initialization and termination of connection between any two nodes. Suppose, a node i is at location L_i and has a transmission range R_i. The node i needs to transfer messages of size *X* bytes to the neighbor node j which is at the location is L_j and have the transmission range R_j. Let the bandwidth available for this communication be *Y* bytes/s. Then, the total time required for the transfer of *X* bytes messages equals $T = X/Y$. When the two nodes communicate, they are in each other's effective range, which is mathematically represented as:

$$|L_i - L_j| \leq min\left(R_i, R_j\right). \tag{3.1}$$

Let $\overrightarrow{v_i}$ and $\overrightarrow{v_j}$ be velocities of both nodes, respectively. The contact duration window between the two nodes can be calculated as follows:

$$T_{cw} = \frac{2 \times min(R_i, R_j) \times cos\,\theta}{|\overrightarrow{v_i} - \overrightarrow{v_j}|}, \tag{3.2}$$

where θ is the angle between the relative velocity $|\overrightarrow{v_i} - \overrightarrow{v_j}|$ and the straight line (distance vector) between the two nodes. Let m and n be the number of messages that both the nodes i and j need to transfer, respectively, then there must be sufficient time available for transfers, specified by:

$$\sum_{i=1}^{m} T_i + \sum_{j=1}^{n} T_j < T_{cw} - \beta \,. \tag{3.3}$$

In above equation, the parameter β is scaling factor that depends on time spent in metadata exchange (control signaling) and link delays. An important question arises here, and can be stated as, "*Given a limited contact duration, which messages should be given priority over others?*" In [3.7] the messages are prioritized on the basis of their size, such that the messages that can fit in the real-time contact window are transferred first, whereas in [3.15] messages with low hop counts are given priority for quicker dissemination in the network. Lindgren *et al.* [3.13] gives priority to those messages whose destinations are most encountered by the neighboring relay node, whereas authors in [3.6] assign priorities to messages whose *utility* contributes to improvement of routing performance. In the next section, we briefly define the ten selected routing protocols and discuss the merits and demerits of each.

3.3. DTN Routing Protocols

For evaluation, we selected the following ten routing protocols: **(a)** *Direct Transmission* [3.16], **(b)** *First Contact* [3.12], **(c)** *Epidemic* [3.14], **(d)** *Wave* [3.17], **(e)** *Life* [3.17], **(f)** *Spray and Wait* [3.18], **(g)** *Spray and Focus* [3.19], **(h)** *PRoPHET* [3.13], **(i)** *MaxProp* [3.15], and **(j)** *Rapid* [3.6]. The aforementioned protocols cover a broad range of scenarios and DTN applications. For instance, the *MaxProp* [3.15] and *Rapid* [3.6] protocols were designed specifically to target a bus-based DTN scenario, where nodes as buses follow mobility at fixed schedules. The protocols, such as *Epidemic* [3.14], *Wave* [3.17], *Life* [3.17], *Spray and Wait* [3.18], and *Spray and Focus* [3.19] in general address the scenarios where nodes mostly follow random mobility patterns. Alternatively, the *Direct Transmission* [3.16], *First Contact* [3.12], and *PRoPHET* [3.13] protocols are designed to target the scenarios where the meeting schedules of nodes can be predicted. Because of diverse properties, the selected set of protocols provide a thorough investigation of the various insights about DTN routing schemes. We begin with the simplest routing protocol of the abovementioned list, and gradually proceed towards more complex models. The discussion gives us an insight into how an increase in complexity of protocols improves the performance for one metric at the cost of another.

3.3.1. Direct Transmission

Spyropoulos *et al.* [3.16] presented the simplest of DTN routing protocols. As the name suggests, the source node directly transmits the message to the destination node without any intermediate relaying. Therefore, at any time, only a single copy of message is present in the network. The benefit of *Direct Transmission* scheme is that it causes minimum overhead due to reduction in messages' copies. However, the source nodes may have to wait for longer periods or indefinitely to make direct contacts with the messages' destinations. Therefore, *Direct Transmission* may experience maximum latency as well as the minimum message delivery ratio. The *Direct Transmission* scheme may be useful for DTN scenarios where nodes' mobility pattern can be precisely predicted. One such application can be bus networks where buses followed fixed schedules for various routes.

3.3.2. First Contact

The *First Contact* scheme [3.12] allows a node to forward a message towards a randomly selected neighbor. After the message is successfully forwarded, the sender node deletes local copy of message. The message may be relayed through several hops before reaching the destination. This makes the *First Contact* a single-copy multi-hop forwarding scheme. A message maintains a list of hops traversed to avoid visiting the same hops again. The *First Contact* protocol does not exhibit optimal performance in terms of message delivery ratio. This is because the randomly selected neighbor may not appear to be a best candidate to forward message towards destination. Therefore, delivery ratio of *First Contact* does not significantly improve, when compared to *Direct Transmission.* Moreover, the *First Contact* scheme experiences higher overhead due to extra transmissions per message.

3.3.3. Epidemic

The *Epidemic* protocol [3.14] is an *uncontrolled replication-based* routing scheme that functions analogous to the way a disease spreads. A node having a message copy is said to be infected. When this node makes contact with another node, the infection is transferred to the other node such that at the end of communication both nodes are having same the infection (similar copies of a message). The *Epidemic* scheme spreads greater number of message copies in the network to improve message delivery probability. However, increasing message copies may cause greater overhead, higher utilization of buffers, and increased network congestion. Therefore, the *Epidemic* protocol is ideal for scenarios that have higher bandwidth and greater buffer storage available. The scenarios where nodes have limited buffer capacity, the *Epidemic* protocol may result in packet drop due to buffer overflows.

3.3.4. Life

The *Life* protocol [3.17] is based on the theory of *Conway's Game of Life* [3.20]. This theory simulates life of a *cell* depending on the number of live cells in the neighborhood. The *cell* represents a node in the network having the message replica. The *Game of Life* theory is applied in the *Life* protocol to control the message flooding. A node may replicate, delete, or keep a message copy depending on the minimum and maximum number of neighbors that have the copy of same message. As the protocol controls message replication by frequently deleting the extra message replicas in the neighborhood, the buffer utilization is reduced. This strategy also improves the message delivery ratio as fewer messages are dropped due to buffer overflows. The *Life* protocol is designed keeping in view the mobility pattern followed by people on sea beaches.

3.3.5. Wave

The *Wave* protocol [3.17] utilizes *tracking lists* to track messages that were recently relayed by a node. The idea is to prevent a node from receiving the same message replica again in short time duration. When a node receives a message, the message entry (such as, message identifier and receiving time) is maintained in the *tracking list.* During message exchange, the sender node transfers the message to the neighbor, but does not remove the message entry from the *tracking list.* This prevents the node from receiving the same message replica within a short time span. Such reductions in message replications minimize the overall overhead. However, decrease in the message replicas also decreases the message delivery probability of the *Wave* protocol, as compared to the *Life* protocol.

3.3.6. Spray and Wait

The *Spray and Wait* protocol *(Binary version)* [3.18] sets a limit on a message's maximum number of replicas in the network to reduce flooding. Every new message is assigned an *L* number of *replication tokens,* which represents the maximum number of replicas a message can have at any time in the network. During the *Spray* phase, a node replicates the message by setting $\lceil L/2 \rceil$ tokens on the message copy the has in buffer, and assigns $\lfloor L/2 \rfloor$ to the message copy sent to the neighbor. A node having a message with token value equal to one enters the *Wait phase* for only that particular message. In the *Wait* phase, the node no longer relays the message, and waits to make a contact only with the message's destination. The *Spray and Wait* protocol experiences minimum overhead due to decrease in per message copies. It also exhibits minimum latency and indicates higher message delivery ratio. This is because, the controlled replication reduces message drop that results in quicker dissemination of messages. Moreover, the reduction in message copies prevents message drops due to shortage of buffer space.

3.3.7. Spray and Focus

The *Spray and Focus* protocol [3.19] utilizes the concept of *utility function*. The utility function quantifies the quality of a node to become a relay for a message. A node with greater and recent interactions with a message's destination is considered to have higher *utility* value for that particular message, and will be more suitable candidate to carry the message. The *Wait phase* in the *Spray and Wait* protocol is replaced by *Focus* phase in the *Spray and Focus* protocol. In the *Focus phase,* a node forwards the message to the neighbor, if and only if the neighbor has higher *utility* value for that message. As compared to the *Spray and Wait,* the *Spray and Focus* protocol experiences lesser latency as the messages do not have to wait in the buffers for indefinite periods to be transferred to the destinations. However, the increase in message transmissions also increases overhead.

3.3.8. PRoPHET

The *PRoPHET* protocol [3.13] calculates the *delivery predictability* for every node in the system. The *delivery predictability* value is quantified by the number of recent interactions of a node with other nodes in the network. Nodes perform the transitive updates of *delivery predictabilities* by sharing routing tables during contacts. A node replicates a message to the neighbor, if and only if the *delivery predictability* of the neighbor is greater than the sender node. This way the *PRoPHET* protocol attempts to reduce the overhead. However, when the network size is large, such as a city-wide DTN network, it may take significant time in building up of *delivery predictabilities*. Therefore, in such cases *PRoPHET* may experience increase in overhead due to higher number of replications. The overhead also results in the increased buffer shortages, which may lead to the reduced message delivery ratio for *PRoPHET*.

3.3.9. MaxProp

The *MaxProp* protocol [3.15] implements the message queue management by splitting the queue into two portions. In the first portion, those messages are prioritized that traversed least number of hops. In the second portion, the messages having the least cost paths towards destinations are given priority. The least cost paths are calculated by using a modified version of *Dijkstra's* algorithm. The *MaxProp* protocol prevents a message from repeatedly visiting the same hop by implementing hop-lists within a message. This significantly reduces the message overhead. The use of acknowledgements deletes the redundant message copies from the nodes' buffers, as a consequence, substantially reducing the message drops due to buffer overflows, and improving overall message delivery ratio.

3.3.10. Rapid

The *Rapid* protocol [3.6] employs the concept of message *utility.* Message *utility* is calculated on the basis of a node's past interactions with a message's destination and the amount of data exchanged during the interactions [3.6]. A message's *utility* is higher, if and only if, the message's replication causes improvement in any of the following routing metrics defined by the authors: **(a)** average delay, **(b)** worst-case delay, and **(c)** number of packets delivered before deadline [3.6]. During a transfer opportunity, the *Rapid* protocol calculates *marginal utility* of all the messages in the routing queue. Marginal utility quantifies the marginal increase in utility of the message, when the said message is replicated. The messages are sorted in the buffer such that the first message to be replicated has the highest *marginal utility*. If network size is large, such as a city-wide network, then the *Rapid* protocol may indicate lower performance. This is because of the time required for building-up of message utilities. Therefore, the large number of replications during such a period may lead to increased overhead.

3.4. Empirical Setups, Results, and Discussion

This section presents the comparisons of the ten selected protocols. The simulations were performed with ONE simulator [3.10] that has rich features for simulating DTNs with a variety of parameters and scenarios.

3.4.1. Performance Metrics

The protocols are evaluated for three performance metrics: **(a)** message delivery ratio, **(b)** latency, and **(c)** overhead. The following subsections illustrate these metrics.

3.4.1.1. Message Delivery Ratio

Message delivery ratio is the percentage of messages delivered successfully. The increased message delivery ratio is the major goal of any DTN routing protocol. Message delivery ratio is calculated as:

$$Message\ Delivery\ Ratio = \frac{1}{M}\sum_{k=1}^{M} R_k\ , \qquad (3.4)$$

where M is total messages created and $R_k = 1$ if message m_k is delivered, otherwise $R_k = 0$.

3.4.1.2. Message Latency

Message latency is the total time spent between message creation and delivery to the destination. The average latencies of messages contribute to the overall latency measure of protocol. A protocol must minimize latency but without compromising message delivery ratio. The latency (in seconds) is given by:

$$Latency\ average = \frac{1}{N}\sum_{k=1}^{N} Receive\ Time_k - Creation\ Time_k\ , \qquad (3.5)$$

where N is the total number of messages received.

3.4.1.3. Overhead

Overhead is the approximate measure of the consumption of bandwidth, energy, and storage by a protocol due to message transmissions. The overhead is calculated as the relative estimate of number of message transmissions:

$$Overhead = \frac{Total\ relayed - Total\ delivered}{Total\ delivered}. \tag{3.6}$$

The overhead ratio indicates extra transmissions for each delivered message.

3.4.2. Mobility Scenarios

The key performance factor for a DTN protocol is how precisely the design assumptions match various mobility patterns. To closely match with real-life scenarios, we selected a city based synthetic mobility model discussed in [3.10]. In this model, the mobile nodes follow various paths on a city's street map given to the simulator as input. We imported the map of City of Fargo, ND, USA from the *OpenStreetMap API* [3.21]. Using the GIS tool OPENJUMP [3.22], the map was post-processed and marked with various locations such as shops, homes, offices, meeting points, universities, and bus stations. Mobile nodes are divided into several groups and assigned various locations on the map. In addition to the map based scenario, the protocols are also evaluated on real-world connectivity traces available at an online repository [3.11]. The simulations under real-traces were performed in an attempt to get better insight into the suitability of a protocol for human mobility scenarios.

3.4.3. Simulation Parameters

The simulation parameters are indicated in Table 3.1. The parameter values are selected keeping in view the real-world scenarios where transfer opportunities and resources are often limited. A few of the DTN protocols utilize additional simulation parameters indicated in Table

3.2. The values of such parameters are selected as proposed by the authors in their respective literatures.

Table 3.1. Commonly used simulation parameters.

Parameter	Value(s)
World size	4250, 3900 m
Simulation time per run	12 h
Radio Interface	Speed:250kbps (2Mbps) Range: 20 m
Message	Size: 500KB-1MB Interval: 1 per min TTL: 500min
Buffer size	10-100MB (maximum buffer size that a node is willing to allocate for message distribution)
Nodes	Buses: 8 Cars: 20 Pedestrians: 72 Total: 100 (any node can be source as well as destination)
Node average Speed	Buses: 10-35 km/h Cars: 10-50 km/h Pedestrians: 0-5 km/h
Mobility	City environment, Real connectivity traces

Table 3.2. Protocol specific parameters.

Protocol	Ref	Parameters
Spray and Wait	[3.18]	$L = 10\ to\ 15\%$
Spray and Focus	[3.19]	$U_{th} = 10\ to\ 90$ $L = 5\ to\ 10\%$ of nodes
PRoPHET	[3.13]	$P_{init} = 0.75\ \beta = 0.25$ $\gamma = 0.98,\ \ T_u = 30$

3.4.4. Comparative Analysis

In this subsection, we present the simulative comparisons of the selected DTN protocols by varying the following: **(a)** buffer capacity, **(b)** message creation rate, **(c)** number of nodes, and **(d)** message size. A few of the protocols exhibited similarities in average performance for the selected routing metrics. Therefore, based on the similarities in the empirical results, we divide the protocols into following groups: ***Group1*** (***a.*** *MaxProp,* ***b.*** *Spray and Wait,* ***c.*** *Spray and Focus,* and ***d.*** *Rapid*), ***Group2*** (***a.*** *PRoPHET,* ***b.*** *Life,* ***c.*** *Wave,* and ***d.*** *Epidemic*), and ***Group3*** (***a.*** *Direct Transmission* and ***b.*** *First Contact).* It is also noteworthy to mention that the simulation results generated in this subsection depend on the selected parameter values for the evaluation. As DTN routing is greatly affected by the network conditions, we may observe different results in scenarios with dissimilar simulation settings.

3.4.4.1. Impact of Change in the Number of Nodes

Simulations are performed to observe the scalability of each protocol and results are obtained for average values of latency, delivery ratio, and overhead. As reflected in Fig. 3.2(a), the *MaxProp* protocol shows the best performance among the other protocols of *Group1*. The *MaxProp* protocol's message acknowledgement mechanism removes the redundant packets from the buffers to allow enough space for new packets. This prevents message drop due to lack of buffer space. Moreover, the delivery ratio of *MaxProp* remains approximately constant. This is because the increase in network load due to the entry of new node is balanced with the establishment of new least-cost paths in the network. Furthermore, *MaxProp* gives priority to the messages with low hop counts which allows the quicker propagation of newer messages, reducing the latency as depicted in Fig. 3.2(b). The overhead for *MaxProp* as indicated in Fig. 3.2(c) is comparatively lower as the message replication is reduced with the increased network connectivity.

The message delivery ratio of the *Spray and Wait* and *Spray and Focus* protocols is roughly the same and constant, as both the protocols set a limit on the maximum number of message copies in the network to control message flooding. The latency of the *Spray and Focus* protocol is less than the *Spray and Wait* protocol as the former employs the utility based forwarding, whereas the latter waits with the last copy of message to make contact with the message's destination. However, the overhead of the *Spray and Focus* protocol is higher due to greater message transmissions as compared to the *Spray and Wait* protocol. Performance of the *Rapid* protocol decreases as the number of nodes increase. The *Rapid* protocol gives priority to messages with greater utilities. If the network area (and size) is large, as we considered in this simulation, nodes will take longer to generate accurate utility values for messages. This will cause uncontrolled message replication initially, which will also result in greater buffer usage. Therefore, the increase in message drop due lack of buffer space will result in lower delivery ratio for the *Rapid* protocol.

It can be further observed from Fig. 3.2 that the routing protocols of *Group2* experience low performance for almost all the three routing metrics. This is due to the flooding strategies utilized by the *Group2* protocols. The increased flooding results in the shortage of buffer spaces in the network. Therefore, nodes drop messages more frequently to accommodate new messages. The *PRoPHET* protocol slightly performs better due to the controlled message replications. The delivery ratio performance of *Life* is better than *Wave* and *Epidemic* as the *Life* protocol deletes the redundant message copies, which are proportionate to the number of replicas in the neighborhood. The *Group3* protocols show the worst performance. Among the *Group3* protocols, the *First Contact* protocol forwards the single copy of message on a randomly selected connection. This may result in message loss if the selected neighbor fails to transfer message towards destination node. The same is the case with the *Direct Transmission* protocol that performs no forwarding unless the neighbor node is the message destination. Therefore, *Direct Transmission* exhibits the lowest message delivery ratio. From Fig. 3.2(c), it is also evident that all protocols follow the similar behavior in terms of overhead, as the increase in number of nodes also increases the number of transmissions per message. However, this is not the case for *Direct Transmission* which has zero overhead because the *Direct Transmission* protocol performs no replications. The *Spray and Wait* protocol indicates best overhead performance due to the limited number of message transmissions.

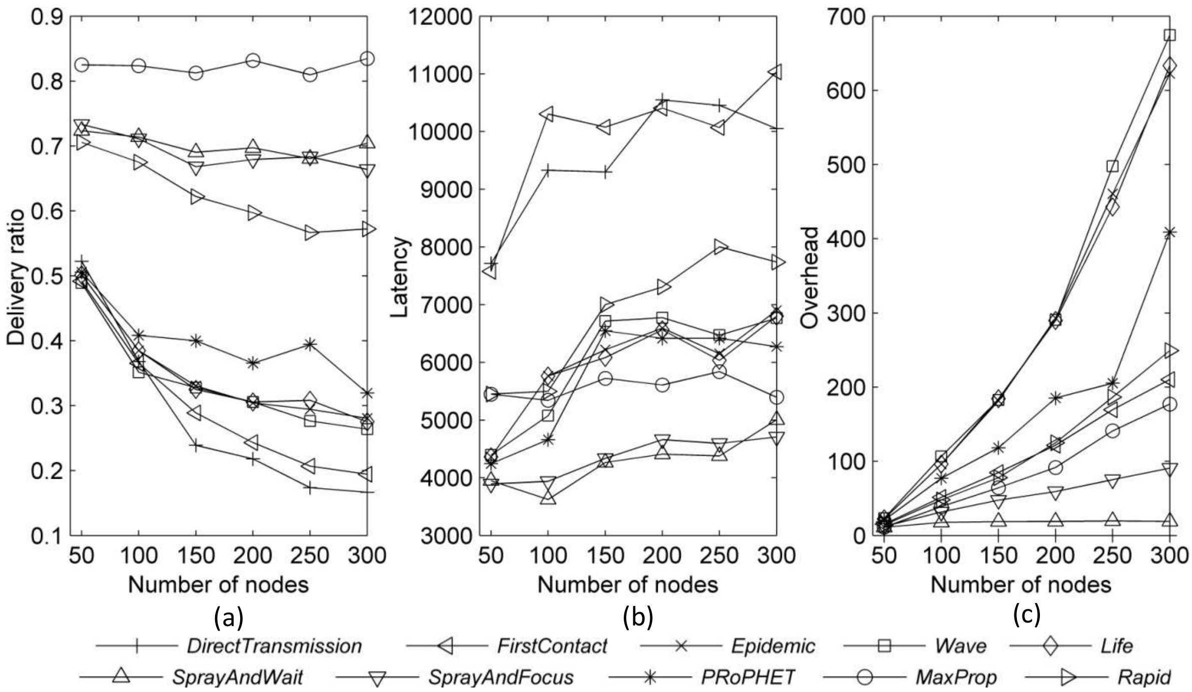


Fig. 3.2. Effect of increasing number of nodes on a protocol's (a) delivery ratio (b), latency, and (c) overhead.

3.4.4.2. Impact of Change in Message Creation Rate

The protocols are evaluated by increasing the network traffic. The message creation rate is varied from one message after ten seconds to one message after 180 seconds (three minutes). As reflected in Fig. 3.3, the *Group1* outperformed the other groups in all the routing metrics. This is because, the *Group1* protocols perform least flooding. When the message creation rate is higher, then the nodes soon exhaust their buffer spaces resulting into the increased message drop rate. The decrease in the message creation rate favors each protocol in terms of message delivery ratio and latency. Among the *Group2* protocols, the *Wave* protocol exhibits better performance due to the utilization of the tracking lists to control flooding [3.17]. The *Group3* protocols experience degradation in message delivery ratio and exhibit increased average packet delay. This is due to the fact that single copy routing strategy implemented by these protocols is less efficient for delivering messages. The overhead (Fig. 3.3(c)) appears to be increasing for the protocols with the decrease in message creation rate. Initially, the message creation rate is higher due to which large numbers of messages are dropped without being relayed, and as a result the overhead is smaller. As the message rate is decreased, there is also an increase in message relaying, which results in an increased overhead in accordance to (3.6). The *Spray and Wait* protocol indicates best performance in terms of overhead due to the fixed number of transmissions per message.

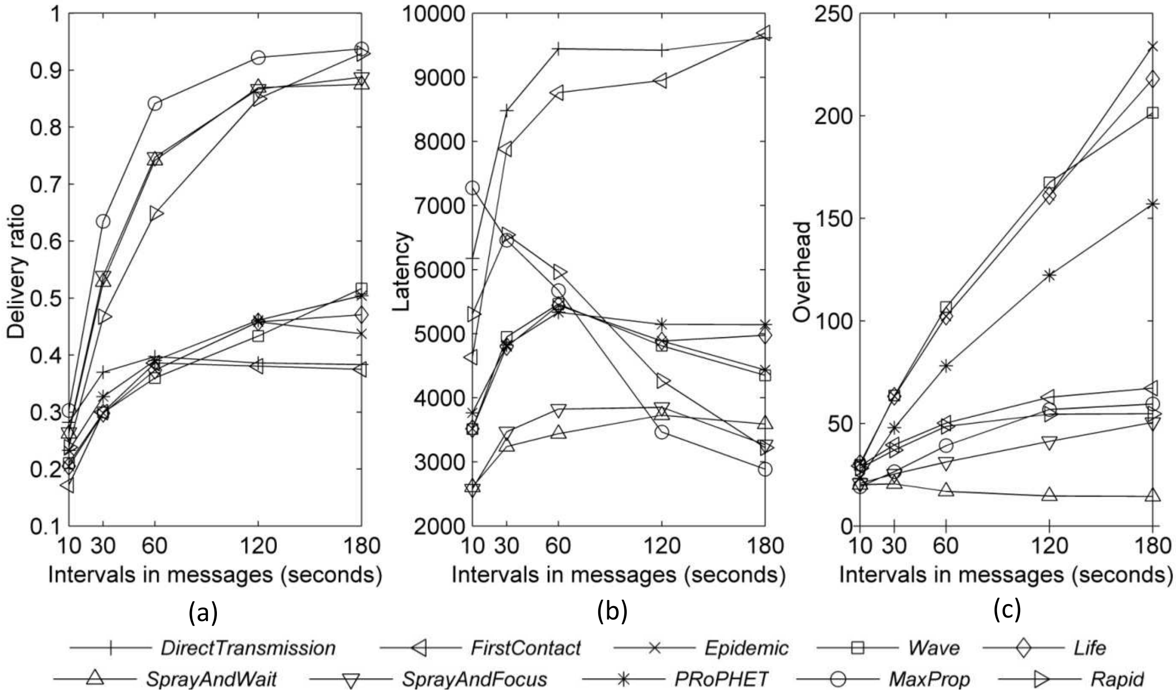


Fig. 3.3. Effect of decreasing the message creation rate on a protocol's (a) delivery ratio, (b) latency, and (c) overhead.

3.4.4.3. Impact of Change in Buffer Capacity

In this experiment, the simulations are performed to understand how protocols behave by varying buffer sizes. As Fig. 3.4(a) indicates, the delivery ratios of *Group1*, *Group2*, and *Group3* protocols initially increase and then achieve a nearly constant value as the buffer size is increased. The reason for such behavior is that initially there is an increased message drop rate due to low buffers, resulting in the low values of delivery ratio. As the buffer size is increased, delivery ratio also improves due to decrease in message drops. However, as reflected from Fig. 3.4(a), raising buffer capacity beyond a certain level does not further contribute to the delivery ratio, as the other factors, such as mobility patterns, buffer management policies, and contact durations, also affect the communications in DTNs. Increasing buffer space favors *Rapid* and *MaxProp* in terms of reduced latency (Fig. 3.4(b)), as both of these protocols require memory space to store metadata information, which is utilized to calculate shortest paths to the destinations. The *Epidemic* protocol does not indicate significant improvement in delivery ratio and latency, despite increase in buffer capacity. The main reasons for such behavior are: **(a)** the buffer management policy of *Epidemic*, which is *First-In-First-Out* (FIFO) and **(b)** limited contact durations among mobile nodes. As long as the contact duration between two nodes is greater than the expected number of messages in transit, FIFO is a reasonable policy. However, if the available contact durations are limited relative to the number of messages, then only a subset of almost similar messages will be transferred on each contact. This will cause the messages at the end of queue to be flushed from buffer due to their life time expiry, resulting in overall decrease in delivery ratio.

The latency of the *PRoPHET* protocol is lesser than other group members as the *PRoPHET* protocol utilizes contacts' history information to relay messages to more appropriate neighbors that have higher delivery predictability for the messages' destinations. Increasing buffer space has interesting impact on overhead ratio (Fig. 3.4(c)). This behavior is in accordance with the (3.6).When the buffer size is small, then fewer messages are delivered as more messages are dropped due to buffer overflows. Therefore, (3.6) will produce a higher overhead value. With increase in buffer capacity, the number of messages delivered also increases, which leads to a decrease in overhead. For *Group3* protocols, the improvement in buffer capacity does not have any significant effect on the delivery ratio. Therefore, the overhead ratio of these protocols remains unchanged.

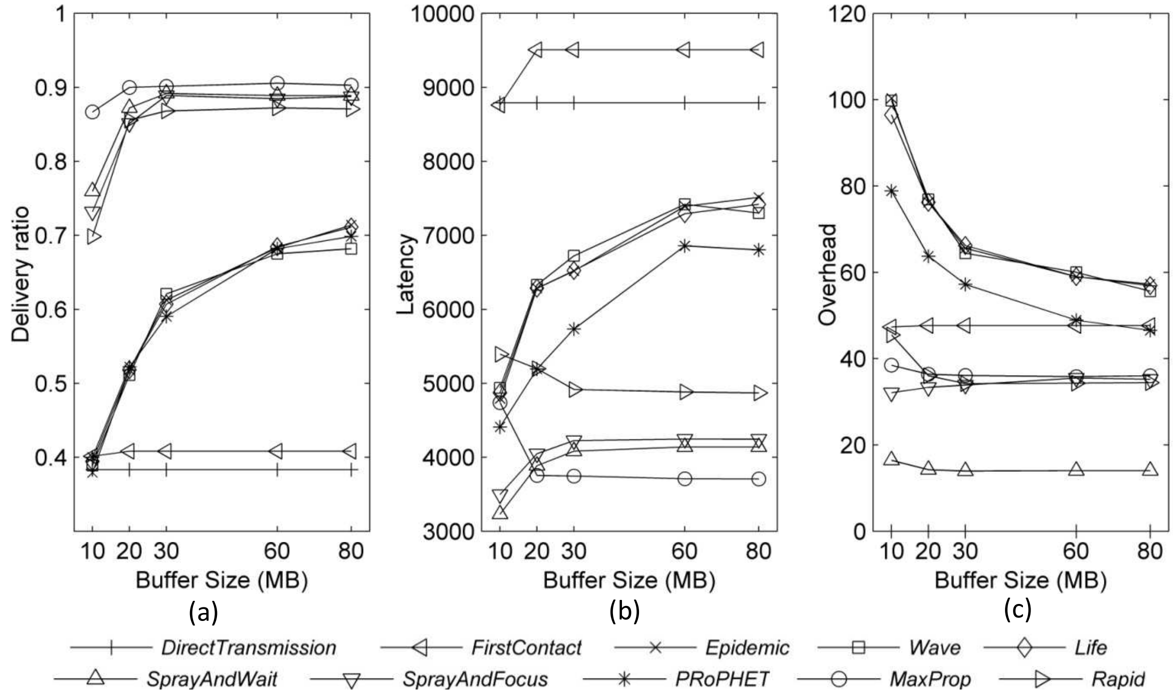


Fig. 3.4. Effect of increasing buffer size on (a) delivery ratio, (b) latency, and (c) overhead.

3.4.4.4. Impact of Change in Message Size

Message size is an important factor for measuring routing performance in DTN environments. As reflected in Fig. 3.5, the increase in message size is almost inversely proportional to the message delivery ratio for all the groups. This is mainly due to the following reasons: **(a)** increase in message size decreases the number of messages exchanged during the limited contact opportunities of mobile nodes, and consequently, most of the messages are dropped as their life-time expires before reaching the destinations, and **(b)** larger messages occupy more buffer space resulting in the buffer shortage for new messages. Therefore, messages are frequently dropped to make room for new messages. The aforementioned point (a) also causes the latency to increase as depicted in Fig. 3.5(b). The *Group1* protocols experience gradual degradation of routing performance for delivery ratio and latency. This is due to the fact that the *Group1* protocols adopt various measures to control flooding, resulting in lesser message drops due to buffers' overflow. In contrast, the *Group2* protocols exhibit a sudden decrease in performance due to their inherent flooding nature. Unlike all the other protocols, the delivery ratio of *Group3's First Contact* protocol (Fig. 3.5(a)) appears to be increasing for message sizes between 100 KB to 500KB. The reason for such an increase is that the *First Contact* protocol deletes the local copy of message after the message is forwarded to the neighbor. Therefore, space is conserved to accommodate more messages, which leads to the decrease in message drop rate. Alternatively, the *Direct Transmission* protocol experiences higher message drops as the nodes' less frequently encounter with the actual destinations of messages. Therefore, the messages are frequently dropped due to life-time expiry.

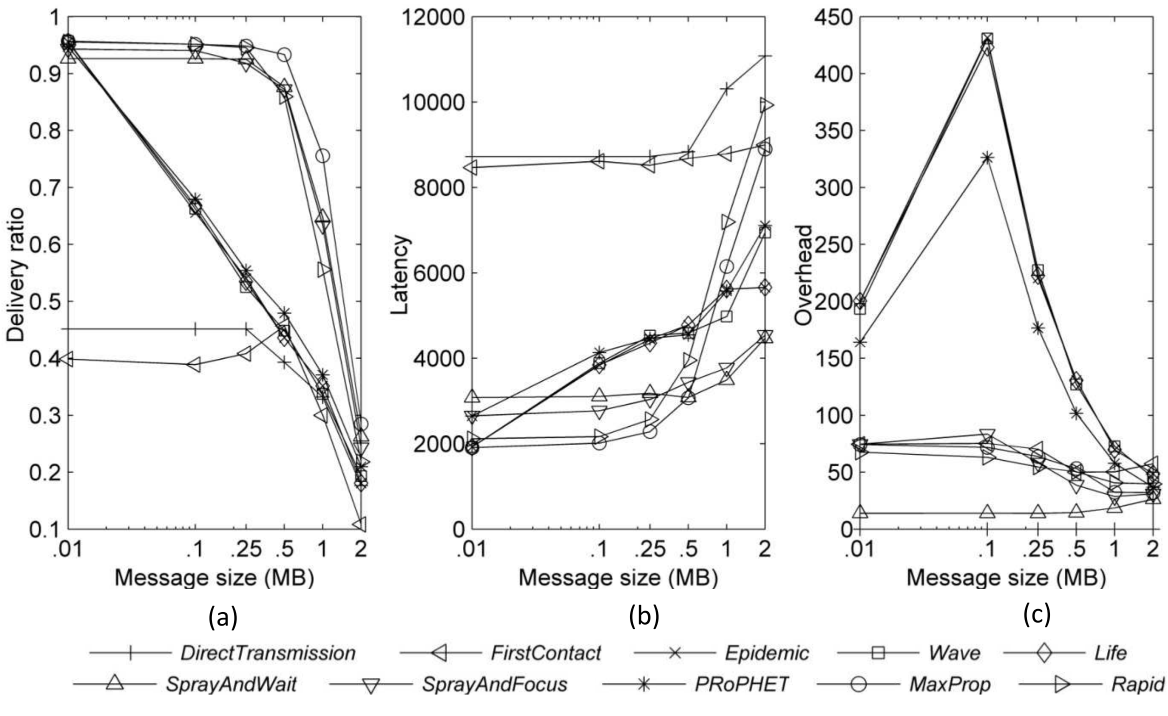


Fig. 3.5. Effect of changing message size on a protocol's (a) delivery ratio, (b) latency, and (c) overhead.

As depicted in Fig. 3.5 (c), the overhead initially increases, and then starts to decline. Initially, message size is smaller and nodes are able to perform more message transmissions, which results in an increase in the overhead. However, the overhead starts decreasing after the message size crosses approximately 100KB. This is due to the decrease in message transfers, as the nodes cannot exchange enough messages in their limited contact opportunities, when the nodes are mobile and messages are of larger sizes.

3.4.4.5. Performance Comparisons with Real Trace Data

In this subsection, the simulations are performed with real-world connectivity traces. The trace datasets were collected under *Haggle project* during Infocom conference and are available at an online repository [3.11]. The parameters used for simulation are: bandwidth: 250Kbps (2Mbps), message size: 500KB–1MB, packet-lifetime: 500 min, number of nodes: 98, and buffer size: 10–100MB.

It can be observed from Fig. 3.6 that *Group1* outperforms the remaining groups in terms of message delivery ratio. The *MaxProp* protocol takes maximum advantage of repetitive mobility patterns of the nodes in calculating least cost paths. As opposed to the synthetic mobility scenario, the delivery ratio of the *Spray and Focus* protocol is greater than the *Spray and Wait* protocol for simulations performed under real-traces. A reason for such behavior is that the *Spray and Focus* protocol keeps on forwarding the packets to the high utility nodes. In this way, the packets are given chance to ultimately reach their destinations in scenarios where transfer opportunities are rare (due to greater inter contact times of nodes). The *Rapid* protocol does not present any significant change in performance for real-trace dataset. The *Rapid* protocol is based on link state routing and it might be challenging to precisely predict the optimal paths between source and destination nodes. Especially, in scenarios when the nodes do not encounter frequently, such as when participants are sitting in different conference sessions for longer durations. Therefore, the *Rapid* protocol experiences greater packet drops due to life time expiry. The latency of the *Spray and Focus* protocol is lesser than the *Spray and Wait* protocol. Moreover, the *MaxProp* protocol exhibits minimum latency among the *Group1* protocols as the

MaxProp protocol more efficiently exploits the repeated mobility patterns in the calculation of least-cost paths.

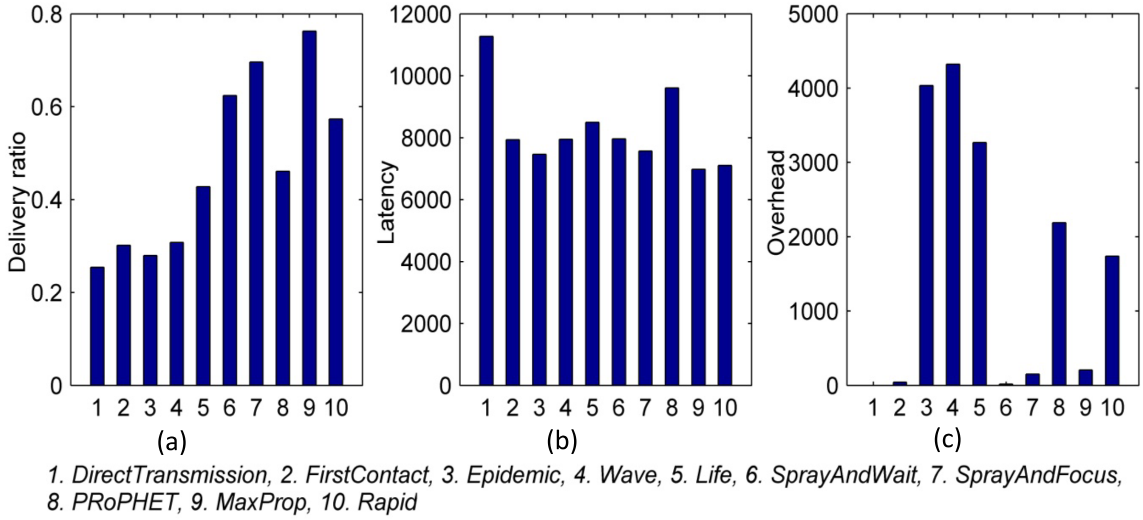


Fig. 3.6. Performance comparison with real connectivity traces for (a) delivery ratio, (b) latency, and (c) overhead.

Among the *Group2* protocols, the *PRoPHET* protocol outperforms the other schemes in terms of delivery ratio. This is primarily due to: **(a)** presence of stationary nodes that are most visited and aid the routing as their delivery predictability improves with time and **(b)** nodes with repetitive mobility act as message relays among various participant groups. However, as it may take longer to encounter a node of higher delivery predictability (such as a stationary node). Therefore, the *PRoPHET* protocol has highest latency among the *Group2* protocols. The *Life* protocol performs better than the *Wave* and *Epidemic* protocols in the conference scenario with low mobility. This is due to the very nature of the *Life* protocol that quickly deletes the message copies and re-gain messages, which helps low buffer occupancy and message circulation. However, good performance of the *Life* protocol comes at the expense of increased overhead (Fig. 3.6(b)). Not surprisingly, the *First Contact* protocol has better message delivery performance than the *Direct Transmission* protocol that has the highest latency among all protocols. The overhead of the *Wave* protocol is greatest among the *Group2* protocols. This is because the *Wave* protocol does not accept a message, if the message's entry is already present in the tracking list [3.17]. This may cause the nodes to occasionally miss the opportunities of becoming message relays. Consequently, there will be slight increase in message drop rate due to life time expiry as messages have to wait longer before being relayed. Therefore, the decrease in "total delivered" parameter of (3.6), will result in the increase of overhead ratio for the *Wave* protocol. The *Rapid* protocol transfers more messages before it can effectively build the metadata, resulting in increased overhead among the *Group1* protocols. The overhead of *Group3* protocols is least as they are single copy message relaying schemes, which perform minimum message transmissions.

3.4.5. Summary of Results and Discussions

Based on the simulation results, it can be concluded that performance of *Group1* protocols remained consistent in all the simulation scenarios, where the *MaxProp* protocol outperformed the rest of the protocols. The primary reason for good performance of the *Group1* protocols is the way these protocols control message flooding. Flooding causes buffer overflows that result in increased message drop rate. Moreover, the *MaxProp* and *Rapid* protocols utilize additional information in the form of metadata to find the shortest paths among sources and destinations, and to delete the messages that have reached the destinations. Alternatively, the *Spray and Wait*, and *Spray and Focus* protocols control flooding by setting a limit on maximum number of message copies in the network. Among the *Group2* protocols, the *PRoPHET* protocol manages to deliver more packets with lesser overhead as compared to the *Life*, *Wave*, and *Epidemic* protocols. This is because the *PRoPHET* protocol controls flooding by replicating a message to the peer, if and only if delivery predictability of peer is higher than the sender. Alternatively, the *Life*, *Wave*, and *Epidemic* protocols rely on maximized flooding that eventually results in higher message drop rate. The performance of *Group3* protocols is lowest as they implemented single copy forwarding, which also minimizes the message delivery chances when nodes' have longer inter-contact times. Therefore, based on the results, we can make following observations:

- **Effect of number of nodes:** Increase in number of nodes facilitates the message forwarding in DTN environments as more nodes are available to serve as relays and to carry message between source and destination. However, such facilitation is at the expense of increased overhead caused by the new nodes. Therefore, the new nodes joining the network do not play a vital role in the improvement of message delivery ratio, as reflected in Fig. 3.2.
- **Effect of message creation rate:** Fig. 3.3 depicts that decrease in the message creation rate improves the delivery ratio due to reduction in message drop rate. However, when the message rate is decreased enough (e.g., one message after every 120 seconds), the delivery ratio for most of the protocols attains a constant value. This constant behavior is due to the other factors involved in message drop, such as: **(a)** message queue sorting policies and **(b)** mobility patterns of nodes. Reduction in message creation rate will reduce latency if and only if the nodes' contact frequency is optimal enough not to cause the messages to wait longer into buffers.
- **Effect of increasing the buffer space:** Increase in buffer capacity of nodes improves message delivery ratio for all the protocols, especially for the *Group2* protocols that utilize flooding based techniques. However, as indicated in Fig. 3.4, even the large sized buffers may not produce 100% message delivery ratio. This is due to the fact that messages may be dropped with the expiry of messages' life time if there are long gaps in nodes' meetings. Another reason is the queue management policies, which may repeatedly favor only a subset of messages during message transfer. For example, the *PRoPHET* protocol gives priority to

messages whose delivery predictability is higher and the *Epidemic* protocol sorts the queue on FIFO basis. This may cause other messages to wait long in buffers and subsequently be dropped due to life-time expiry. Therefore, in current scenario, Fig. 3.4 presents an estimate of upper bound in delivery ratio when the buffer capacity is assumed to be infinite.

- **Effect of change in message size:** When message size is smaller and buffer space is enough to accommodate new messages, the nodes are able to exchange more messages during a limited contact interval, resulting in an increased delivery ratio. As message size is increased, nodes have to wait longer for the occurrence of contacts of greater durations. This causes the message drop due to life-time expiry and increases the latency as indicated in Fig. 3.5.
- **Effect of change in bandwidth:** We also performed simulations by varying the bandwidth. Results indicated that if the bandwidth is higher, then there is an increase in message drop rate due to buffer overflows, but decrease in message drop due to message life time expiry. Alternatively, if bandwidth is less, then fewer messages are exchanged among nodes. Therefore, there is less message drop due to buffer overflow, but higher message drop due to message life time expiry. Therefore, in both the cases we achieved roughly the same results for message delivery ratio.

Based on our findings, we are now in a position to rank protocols depending upon their performance consistency under the parameters selected for simulation. Table 3.3 presents the numerical ranking of the protocols for message delivery ratio. The overview of results and our recommendations on the usage of protocol for a particular scenario is summarized in Table 3.4.

Table 3.3. Protocols ranking based on message delivery ratio. The protocol with lowest score is given top ranking.

Protocol	Message Delivery Ratio				Overall score	Rankings
	Network size	Buffer impact	Message rate	Message size		
Direct Transmission	10	10	10	9	39	10
First Contact	9	9	10	10	38	9
Epidemic	7	6	8	8	29	8
Wave	8	8	6	6	28	7
Life	6	5	7	7	25	6
Spray and Wait	2	2	3	2	9	2
Spray and Focus	3	3	2	3	11	3
PRoPHET	5	7	5	5	22	5
MaxProp	1	1	1	1	4	1
Rapid	4	4	4	4	16	4

Table 3.4. Overview of results.

Protocol	Delivery Ratio	Buffer Utilization	Energy Efficiency	Complexity	Suggested Scenario
Direct Transmission	Very low	Very low	Low-medium	Very low	Deterministic mobility scenarios
FirstContact	Very low	Very low	Low	Low	Random mobility /emergency scenario
Epidemic	Low	Very high	Very low	Low	Random mobility /emergency scenario
Wave	Medium	Medium	Medium	Medium	Partial mobility / limited area scenario
Life	Medium	Medium	Medium	Low-medium	Partial mobility / limited area scenario
Spray and Wait	High	Low	High	Low-medium	Heterogeneous scenarios
Spray and Focus	High	Low-medium	High	Medium-high	Heterogeneous scenarios
PRoPHET	Medium	Medium-high	Medium	High	Community based / predictable mobility
MaxProp	Very high	High	Very high	Very high	Heterogeneous scenarios
Rapid	Medium-high	High	Medium-high	Very high	Limited vehicular mobility scenarios

3.5. Proposed Routing Models

In this section, we propose three routing models based on enhancements in the replication strategies of the following three protocols: **(a)** *Epidemic* [3.14], **(b)** *Spray and Wait* [3.18], and **(c)** *PRoPHET* [3.13]. The reason for selection of only these three protocols is that these protocols are most cited in the DTN literature for evaluations, such as [3.3], [3.5]–[3.9], [3.15], [3.17], as the aforementioned protocols are known to exhibit consistent performances in many different type of DTN scenarios. Alternatively, the *MaxProp* and *Rapid* protocols were specifically designed for bus-based DTN scenarios, and the *Wave* and *Life* protocols were developed for scenarios where the nodes were mostly static. (We did not consider single copy routing schemes in our evaluations because of their low performance.) Therefore, to test the performance of our proposed enhanced schemes, we designed the scenarios that closely matched with the original scenarios in which the abovementioned protocols were evaluated. Moreover, to favor the *Epidemic* and *Spray and Wait* protocols, we introduced random mobility nodes in our defined scenario. Similarly to accommodate the *PRoPHET* protocol, our scenario also contains grid-based communities. The message replication in the proposed schemes is made adaptive to the varying network conditions. The source code of the proposed protocols is made available on this link [3.23]. In the following subsections we illustrate the proposed schemes.

3.5.1. Enhanced Epidemic Scheme

The *Epidemic* protocol performs large scale flooding to increase messages' delivery probability. However, flooding cause network congestion and buffer shortage, resulting in an increased message drop and overhead. In realistic scenarios, the nodes' densities at various network regions do not stay uniform as the network topology varies with time. Therefore, message replication should be adjusted depending upon a node's frequency of interactions with specific destinations. The same concept is applied here in the proposed enhancement of the *Epidemic* protocol.

The new technique *e-Epidemic* attempts to decrease message flooding in cases when specific set of destinations are repeatedly encountered. In *e-Epidemic*, when a node i comes into contact with a neighbor node j, both the nodes exchange metadata information about their recent interactions with other network nodes. This information is quantified as:

$$E_i^d = \frac{1}{T_c - T_i^d} \times \sum_k C_i^d(k). \quad \text{for } k \in \mathbb{N} \tag{3.7}$$

In above equation, $\sum_k C_i^d(k)$ represents interaction history of node i with the message's destination d, the parameter T_c is the current time, and T_i^d is the last interaction time between the node i and the node d. The last interaction times are maintained as local timers at the nodes, as was performed in [3.19]. Equation (3.7) states that a node i will "forward" a message to a neighbor node j, if and only if, the node j's interaction rate (E_j^d) with destination d is higher and more recent than node i's interaction rate (E_i^d). Otherwise, node i will simply "replicate" the message. This behavior is different from the original *Epidemic* protocol where a message is always replicated, and as a result the *Epidemic* protocol exhibits low performance in scenarios with limited buffer size. Fig. 3.7 reflects the performance improvement achieved by *e-Epidemic* for delivery ratio and overhead. The decreased number of copies per message reduces the number of transmissions per message, which results in the reduction of overhead. However, decreased number of copies per message also increases the overall latency (Fig. 3.7(b)). This is because of the increased hop count per message. The delivery ratio is also further improved by introducing passive message acknowledgements for already delivered messages. The acknowledgements help in clearing up buffers from redundant message copies. Therefore, there will be less message drop due to lack of buffer space.

3.5.2. Adaptive Multi-Copy Spray (AMS)

The *Spray and Wait* protocol controls flooding by setting a limit on the maximum number of message copies in the network. When a message is created, it is assigned a specific number of "forwarding tokens". When the message is replicated, the sender node splits the tokens into half, keeping half for itself and giving the other half to the neighbor node.

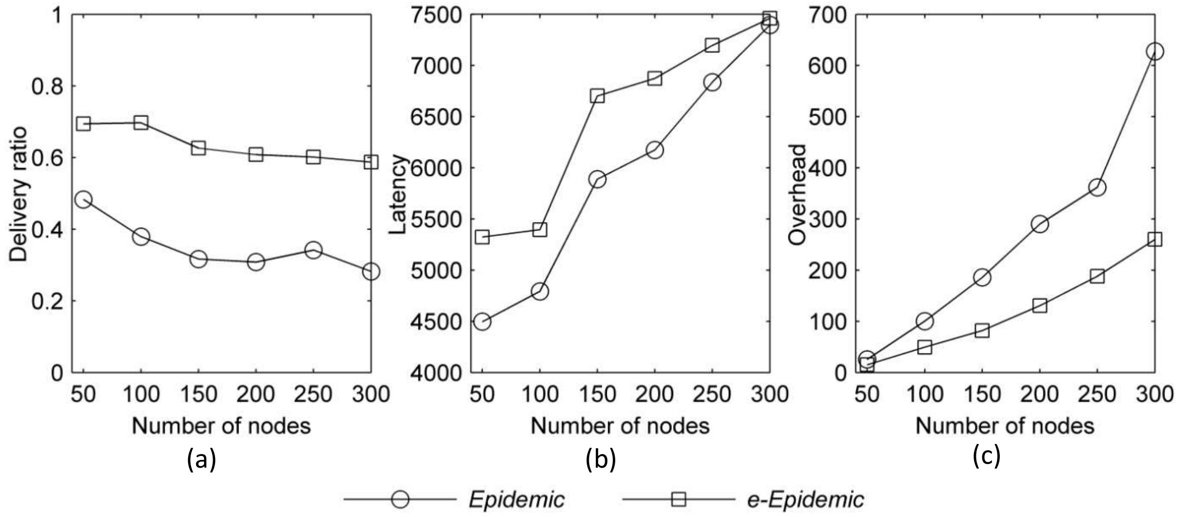


Fig. 3.7. Performance comparison between Epidemic and E-Epidemic protocol.

Although splitting the message tokens into half reduces the overhead, a more optimal splitting based on varying network conditions may lead to further reduction in overhead and improved delivery ratio. Therefore, we propose an enhancement in the *Spray and Wait* protocol such that the token splitting is made adaptive, depending on a node's interaction frequency with other nodes in the network. The proposed technique, *Adaptive Mutli-Copy Spray (AMS)* utilizes the following equation to calculate a node's interaction history:

$$E_i^d = \left(\sum_k C_i^T(k) + \alpha \times \sum_l C_i^d(l) \right) \times \frac{T_c}{2T_c - T_i^d}, \tag{3.8}$$

where E_i^d is the interaction rate. In (3.8), the parameter $\alpha = \sum_m C_i^d(m) \,/\, \sum_n C_i^T(n)$ is the adaptive weight factor representing the fraction of a node's interactions with a destination d, and is added to the total interactions $\sum_k C_i^T(k)$ of the node i. The term $T_c/(2T_c - T_i^d)$ on the right hand side of (3.8) ensures that higher weightage should be given to more recent contacts. A newly created message m_k at node i is initially assigned L forwarding tokens, which is the same number of forwarding tokens used by the original versions of the *Spray and Wait* and *Spray and Focus* protocols ($L \approx 10 - 15\%$ of the total number of nodes). From Fig. 3.8, we observe that the *AMS* protocol achieves higher delivery ratio at roughly same value of L, with a network size of hundred nodes.

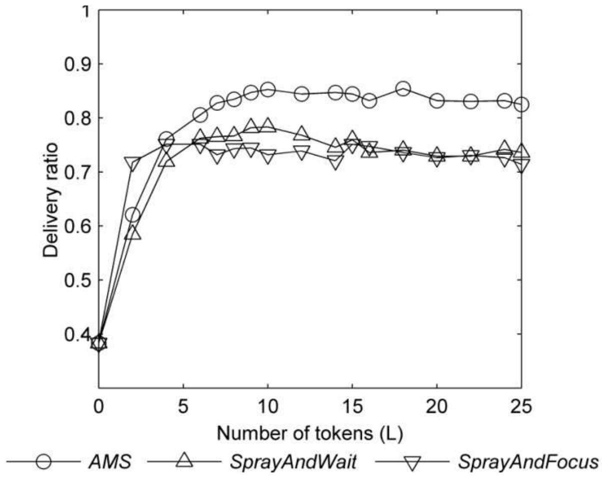


Fig. 3.8. The max delivery ratio in AMS, Spray and Wait, and Spray and Focus is achieved with L=10–15.

Let E_i^d and E_j^d be the interaction histories of node i and j, respectively, with a message's destination d. Suppose, a message m_k has L_k forwarding tokens. When node i transfers m_k to node j, the token splitting is performed, such that node i keeps L_k^i tokens and gives L_k^j tokens to node j according to the following equations:

$$\text{Node } i \text{ tokens} \quad L_k^i = \left(L_k \times \frac{E_j^d}{E_i^d + E_j^d} \right), \tag{3.9}$$

$$\text{Node } j \text{ tokens} \quad L_k^j = \left\lceil L_k \times \frac{E_i^d}{E_i^d + E_j^d} \right\rceil. \tag{3.10}$$

The above two equations indicate that a node with lesser number of interactions with a message's destination will receive greater number of tokens and vice versa. The proposed approach is intended to improve the message delivery through those nodes which have less frequently interacted with the destination d, by increasing the per message replications performed by such nodes.

The simulation results in Fig. 3.9 indicate that the proposed *AMS* scheme performs better than *Spray and Wait* and *Spray and Focus* in terms of delivery ratio and overhead because of the adaptability in the message replication.

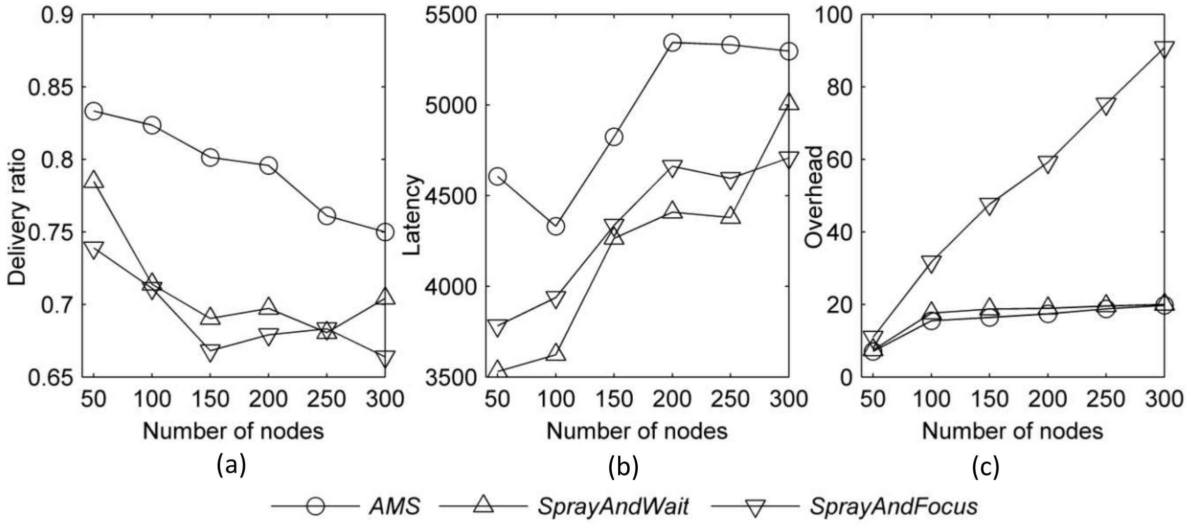


Fig. 3.9. Performance comparison of AMS with Spray and Wait, and Spray and Focus.

However, reduced overhead is at the expense of higher latency. This is because the greater numbers of messages are replicated through nodes that less frequently interacted with messages' destinations. The message drop rate due to buffer shortage is controlled by introducing passive message acknowledgements that further improve the delivery ratio performance.

3.5.3. Adaptive Source Token Multi-Copy Spray (ASTMS)

As illustrated in the *AMS* protocol, a message's initial token value is statically assigned (Section 5.2). In the proposed *ASTMS* scheme, the initial token assignment is made adaptive to a node's varying interaction history in the network. The idea presented here is mathematically formulated as follows:

$$L_k^i = \left\lceil (1 - E_i^d) \times L \right\rceil, \tag{3.11}$$

where the parameter L_k^i is the token value generated for the message m_k at the node i. The interaction rate E_i^d is given by:

$$E_i^d = \frac{C_i^d \times T_c}{C_i^T \times (2T_c - T_i^d)}. \quad (3.12)$$

In above equation, the interaction rate E_i^d depends on the fraction of recent contacts between node i and destination d, whereas C_i^T is the total interaction rate of i, in the whole network. The parameter L is similar to the one used in the previous subsection and is considered as an upper bound to the maximum value of forwarding token. Equation (3.11) indicates that if the node i more frequently interacts with the destination d, then smaller token value L_k^i will be generated for the message m_k at node i. Therefore, the initial token value is dependent on a node's interaction frequency with a specific destination set. Based on the proposed approach, we introduce adaptive token assignment mechanism in the *PRoPHET* protocol. The *PRoPHET* protocol performs message replication if and only if the delivery predictability of the current message is comparatively higher at the neighbor node. However, the *PRoPHET* protocol sets no limit to the maximum number of message replicas. This may cause an increased overhead and message drop rate in the resource constrained network environments. The aforementioned problem is addressed in the proposed *ASTMS* scheme in the following ways: **(a)** by introducing adaptability in initial token assignment, **(b)** by setting a limit on maximum number of message replicas, and **(c)** by introducing adaptability in token splitting during message replication. All such enhancements depend on a source node's interaction history with the message's destination. Algorithm 3.1 presents the pseudo-code for the *ASTMS* scheme. A message created by an application (App) is assigned the initial token value and stored in the message queue at the node i (Line 1). A message with token value greater than one will be replicated, if and only if the interaction history of neighboring node is greater than the current node (Line 3 and Line 4). The token splitting is performed in Line 5, such that, a node with more frequent interactions with the message destination will get higher number of tokens. A message with token value equal to one will be forwarded towards a neighbor with highest interaction value E_k^d (Line 9 and Line 10). The same process will be repeated for other messages in the queue.

Algorithm 3.1. Pseudo-code for ASTMS scheme

1: $m_k \leftarrow App$; $L_k^i = \lceil (1 - E_i^d) \times L \rceil$; $M_i = M_i \cup \{m_k\}$
2: **for each** $m_k \in M_i$ **do**
3:     **if** $L_k > 1$ **then**
4:         **if** $E_j^d > E_i^d$ **then**
5:             $L_k^j = L_k \times E_j^d / (E_i^d + E_j^d)$
6:         **end if**
7:         *Replicate* m_k to ***j***
8:     **else**
9:         Find a neighbor *k* with highest E_k^d
10:         *Forward* m_k to ***j***
11:     **end if**
12: **end for**

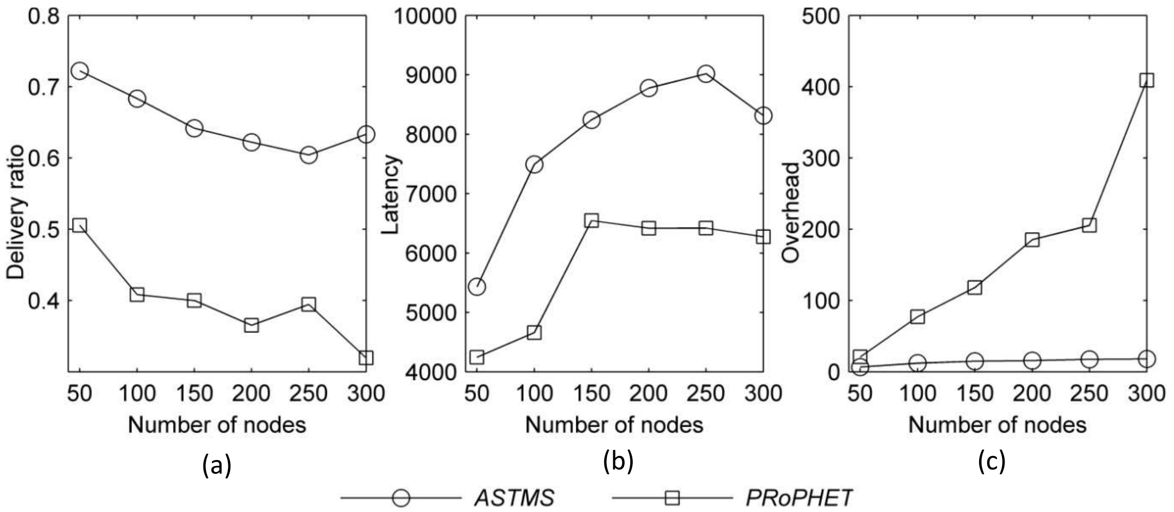


Fig. 3.10. Performance comparison of ASTMS with the PRoPHET protocol.

Fig. 3.10 indicates the improved performance of proposed scheme in comparison to the *PRoPHET* protocol. The reduction in message replications by setting a limit on maximum number of message copies results in less utilization of buffer space as well as decreases the message drop rate. Therefore, message delivery ratio is significantly improved. Moreover, the decrease in message copies improves the overhead, as per message transmissions are also minimized. However, the reduced overhead is at the expense of increased latency. This is because, with the decrease in message replications, the messages may have to wait longer to be delivered to the destinations.

3.6. References

[3.1] Delay-Tolerant Networking Research Group (DTNRG). [Online] http://www.dtnrg.org/wiki, accessed July, 2014.

[3.2] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss. Delay-Tolerant Networking: “An Approach to Interplanetary Internet,” *IEEE Communication Magazine,* vol.41, no.6, pp.128-136, 2003.

[3.3] T. Spyropoulos, R. N. B. Rais, T. Turletti, K. Obraczka, and A. Vasilakos, “Routing for disruption tolerant networks: taxonomy and design,” *ACM Journal Wireless Networks,* vol.16, no.8, pp.2349-2370, 2010.

[3.4] T. Spyropoulos, T. Turletti, and K. Obrazcka, “Routing in Delay Tolerant Networks Comprising Heterogeneous Populations of Nodes,” *IEEE Transaction on Mobile Computing*, vol. 8, no. 8, pp. 1132-1147, Aug. 2009.

[3.5] O. Khalid, S. U. Khan, J. Kolodziej, L. Zhang, J. Li, K. Hayat, S. A. Madani, L. Wang , and D. Chen, “A checkpoint based message forwarding approach for opportunistic communication,” *European Conference of Modeling and Simulation,* 2012.

[3.6] A. Balasubramanian, B. Levine, and A. Venkataramani, “Replication Routing in DTNs: A Resource Allocation Approach,” *IEEE/ACM Transactions on Networking,* vol.18, no.2, pp.596-609, 2010.

[3.7] G. Sandulescu and S. Nadjm-Tehrani, “Opportunistic DTN routing with window-aware adaptive replication,” in *Proc. 4th Asian Conf. on Internet Engin*, 2008, pp.103-112.

[3.8] Y. Cao and Z. Sun, “Routing in Delay/Disruption Tolerant Networks: A Taxonomy, Survey and Challenges,” *IEEE Communications Surveys and Tutorials*, vol. 15, no. 2, pp.654-677, 2013.

[3.9] A. G. Voyiatzis, “A Survey of Delay- and Disruption-Tolerant Networking Applications,” *Journal of Internet Engineering*, vol. 5, no. 1, pp.1-331, 2012.

[3.10] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE simulator for DTN protocol evaluation,” in *Proc. of 2nd International Conference on Simulation Tools and Techniques*, 2009, doi.10.4108.

[3.11] Infocom06 connectivity traces on CRAWDAD website. [Online] http://crawdad.cs.dartmouth.edu/meta.php?name=cambridge/haggle#N100C4, 2013.

[3.12] S. Jain, K. Fall, and R. Patra, “Routing in a delay tolerant network,” in *Proc. of Conference on applications, technologies, architectures, and protocols for computer communications*, 2004, pp.145-158.

[3.13] A. Lindgren, A. Doria, and O. Schelén, “Probabilistic routing in intermittently connected networks,” in *SIGMOBILE Mobile Computing and Communications Review*, vol.7, no.3, pp.19-20, 2003.

[3.14] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” *Technical Report CS-200006*, *Duke University*, 2000.

[3.15] J. Burgess, B. Gallagher, D. Jensen, and B. Neil Levine, “MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks,” in *Proc. of IEEE Infocom*, April, 2006, pp.1-11.

[3.16] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Single-copy routing in intermittently connected mobile networks, in *Proc. of Sensor and Ad Hoc Communications and Networks*, 2004, pp.235-244.

[3.17] J. Ott, A. Keränen, and E. Hyytiä, “BeachNet: Propagation-based Information Sharing in Mostly Static Networks,” in *Proc. of ACM ExtremeCom*, 2011.

[3.18] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks,” in *Proc. of ACM WDTN*. 2005, pp.252-259.

[3.19] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility,” in *Proc. of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshop*, 2007, pp. 79-85.

[3.20] M. Gardner, “The fantastic combinations of John Conway’s new solitaire game life,” *Scientific American*, pp.120-123, 1970.

[3.21] *OpenStreetMap (OSM) API website.* 2012. http://www.openstreetmap.org/.

[3.22] *OPENJUMP GIS Software*, [online] http://www.openjump.org/, accessed on July, 2013.

[3.23] O. Khalid, Source code http://www.ndsu.edu/pubweb/~okhalid, 2013.

[3.24] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, and A. Qayyum, “Optimized Link State Routing Protocol (OLSR),” RFC 3626, IETF Network Working Group, October 2003.

[3.25] C. Perkins, E. Belding-Royer, and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” RFC 3561, IETF Network Working Group, July 2003.

[3.26] D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks,” Imielinski and Korth, Eds., Mobile Computing, Vol. 353, 1996.

4. FORECAST AND RELAY: A MESSAGE ROUTING SCHEME FOR OPPORTUNISTIC MOBILE NETWORKS

This paper is submitted to *IEEE Communication Letters*. The authors of this paper are Osman Khalid, Rao Naveed Bin Rais, Nasir Ghani, and Samee U. Khan.

4.1. Overview

The increasing integration of communication technologies in mobile devices, such as Bluetooth and 802.11/WiFi has made it possible for the mobile nodes to opportunistically connect and form on-the-fly ad hoc networks, known as *Opportunistic Mobile Networks* (OMNs) [4.1]. The OMNs are usually deployed to establish communications in challenging environments, where nodes suffer from frequent disconnections and are not supported by any fixed infrastructure. A few applications of OMNs include [4.2]: (i) battle-field networks, (ii) wild-life monitoring, (iii) emergency response systems, (iv) vehicular transportation bus networks, and (v) mobile social networks [4.1]. The network in such scenarios frequently changes topology, and in certain extreme cases, there may not be an end-to-end path available between a source and a destination. Therefore, the traditional mobile ad hoc network routing protocols, such as AODV [4.3] are inapplicable in such scenarios. Message routing in the OMNs is difficult, as the nodes have little information pertaining to the state of the network that has a time evolving topology. The nodes must rely on opportunistic contacts to store-carry-forward messages towards the destination. The messages may be delivered or dropped due to message life-time expiry (TTL) or network congestion, yielding a best-effort delivery service.

4.1.1. Motivation

During an opportunistic contact, a node must decide whether or not to forward the data to the encountered node. The message forwarding decisions within the OMNs are typically guided by the desire to reduce the number of message replicas within the network that consequently conserves buffer space and the bandwidth. However, due to the temporal variations within the topology, identifying multiple end-to-end disjoint paths for messages is often very difficult.

Several works, such as [4.4]–[4.8] have been presented on the routing/forwarding mechanisms for the OMNs. However, there is no consensus on which approach best suits a given scenario or application. Among all the routing strategies, the single-copy forwarding schemes are considered to be the most resource conservative, as such approaches are designed to forward only a single copy of the message within the network [4.6]. However, such approaches suffer from delay and reduced message delivery, as single message copy may have to wait longer. In some cases, the message may never reach the destination. The flooding-based approaches spread multiple replicas of a message within the network [4.7]–[4.9]. A higher number of message replicas improves the message delivery probability, but at an increased expense of network resources. The authors in [4.10] set a limit on the number of replicas per message that lowered the overhead, but increased the message delay. Lindgren *et al.* [4.5] proposed a probabilistic message replication approach, named as the *PRoPHET*, where a node replicates a message to a neighbouring node, if and only if, the neighbouring node has more frequently encountered with the destination. However, the *PRoPHET* protocol is not mobility-cognizant and sets no limit on the number of message replicas. To address network overhead, the techniques proposed in [4] and [4.7] restrict a message replica from entering a node's buffer for a certain period of time. However, such a restriction results in increased delay as messages are restrained from quickly spreading within the network.

4.1.2. Contributions

In light of the above discussion, there is still a pressing need to develop resource conserving routing solutions for the OMNs that must exhibit better message delivery rates with reduced overhead. This chapter addresses the critical area of resource efficiency in the OMN routing, and exploits the nodes' mobility patterns to control message replicas within the network. Several of the previous studies reveal that humans follow repetitive schedule of meetings at similar places and times, and that the human mobility is predictable and follows a power-law distribution [4.11]. The aforementioned fact is further endorsed by Song *et al.* that the human mobility is 93% predictable [4.12]. Therefore, our proposed Forecast and Relay (FAR) scheme learns from the nodes' temporal contacts and mobility patterns, and forecasts the future contact opportunities among the nodes. The proposed scheme also controls message flooding by forwarding the messages to only those nodes that increase the message delivery likelihood.

The *FAR* protocol can be efficiently applied in real-life applications, such as the transfer of delay-tolerant data among various parts of a city. Simulation results with real mobility traces and synthetic mobility indicate that the *FAR* protocol exhibits better performance in terms of delivery rate and overhead compared to the other schemes, such as *PRoPHET* [4.5], *Epidemic* [4.8], *Random* [4.8], and *Wave* [4.7]. To the best of our knowledge, this is the first effort on utilizing time-series forecasting in the OMNs to develop efficient route prediction.

The rest of the chapter is organized as follows. The network model and assumptions are discussed in Section 4.2. In Section 4.3, we present the *FAR* protocol design, and simulation results are discussed in Section 4.4.

4.2. Network Model and Assumptions

We consider a hybrid network environment consisting of mobile nodes and fixed wireless Access Points (AP) as shown in Fig. 4.1. Two nodes are able to transfer messages only when they are in each other's transmission range. During a message transfer, a sender node may or may not replicate the message on the neighboring node. As there are no end-to-end connectivity paths among mobile nodes, a mobile host can either directly deliver the message to the destination, or through the intermediate relays. Moreover, the mobile nodes have limited storage and transfer bandwidth. We further assume that some of the nodes, such as buses, follow schedules in the mobility. Fig. 4.1 depicts a real-world scenario where people visit bus stops on a daily basis to reach their work place, and buses travel on predefined routes with specific route timings. Such repetition in the mobility patterns can be useful for opportunistic message relaying in different parts of a region. Table 4.1 lists the most frequent notations used in this chapter.

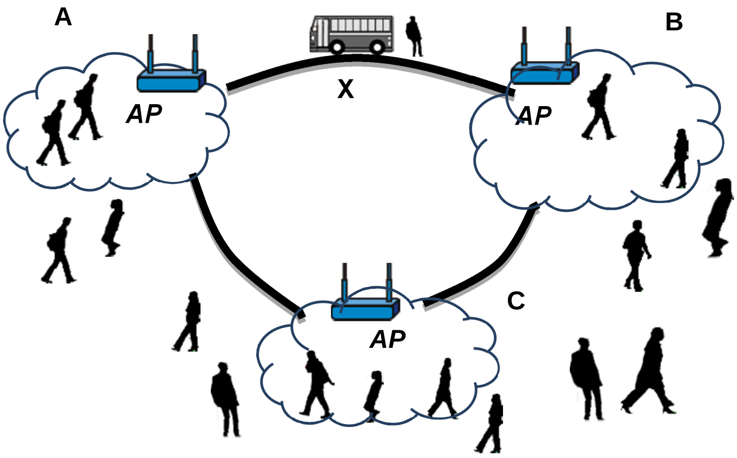


Fig. 4.1. Mobile carries are exchanging messages on making opportunistic contacts.

Table 4.1. Notations and their definitions.

Notation	Definitions
N	Set of nodes
s	Source node
d	Destination node
r	Relay node
t	Current time interval
$C_{ij}(t)$	Meeting quality between nodes i and j at time t
$F_{id}(t)$	Forecasted meeting quality between i and d at time t
ϕ	Time series smoothing constant

4.3. FAR Protocol Design

We consider an OMN, with *N* number of nodes. When a node *i* meets with another node *j* at time interval *t*, both of the nodes record the quality of the meeting, represented by $C_{ij}(t)$. In the *FAR* protocol, the duration of the contact between any two nodes quantifies the meeting quality. Therefore, the greater the contact duration between two nodes, the greater the message exchange probability and higher the meeting quality. With the passage of time, a node *i* will be cumulating a bivariate time-series data consisting of a contact time and contact duration for every other network node. Suppose that a source node *s* has a message *m* to be delivered to a destination node *d*. If *s* makes a direct contact with *d*, then the message will be transferred to the destination without any intermediate relaying. Otherwise, *s* decides whether or not to replicate the message on a relay node *r* that comes in range. Such a decision is performed by *s* on the basis of the following:

$$F_{sd}(t+1) = \phi \cdot C_{sd}(t) + (1-\phi) \cdot F_{sd}(t). \tag{4.1}$$

In the above equation, the parameter $0 \leq \phi \leq 1$ is the time-series smoothing constant, $C_{sd}(t)$ is the meeting quality of *s* with *d* until time *t*, $F_{sd}(t)$ is the current forecast, and $F_{sd}(t+1)$ is the future forecast of meeting the quality of *s* with *d*. Equation (4.1) implies that the message *m* will be replicated by s on r if and only if *r* has a better forecast of meeting quality with *d*. We represent such a phenomenon by $F_{rd}(t+1) > F_{sd}(t+1)$. As mobile devices are limited in memory and processing, it is impossible for the mobile nodes to store unlimited time-series data of the past interactions. Therefore, we set a limit on the maximum number of entries stored per node in the form of a sliding time window, $1 \leq t \leq \omega$, where the entry at $t = \omega$ represents the latest meeting. More recent entries within the range $[1, \omega]$ must be given higher weightage than the other entries to ensure information freshness and accuracy. Therefore, we assign progressively decreasing weights to the older entries. Substituting the value of $F_{sd}(t) =$

$[\phi \cdot C_{sd}(t-1) + (1-\phi) \cdot F_{sd}(t-1)]$ in (4.1) we obtain:

$$F_{sd}(t+1) = \phi \cdot C_{sd}(t) + (1-\phi) \cdot [\phi \cdot C_{sd}(t-1) + (1-\phi) \cdot F_{sd}(t-1)]\,. \tag{4.2}$$

Re-substituting the value of $F_{sd}(t-1)$, and solving recursively, we get:

$$\begin{aligned} F_{sd}(t+1) = \phi \cdot C_{sd}(t) + \phi \cdot (1-\phi) \cdot C_{sd}(t-1) + \phi(1-\phi)^2 \cdot F_{sd}(t-2) + \cdots \\ + (1-\phi)^{\omega} \cdot F_{sd}(0). \end{aligned} \tag{4.3}$$

The above equation can also be written as:

$$F_{sd}(t+1) = (1-\phi)^{\omega} \cdot F_{sd}(0) + \sum_{k=1}^{\omega} \phi \cdot (1-\phi)^{\omega-k} \cdot C_{sd}(k). \tag{4.4}$$

In the above equation, each entry for the meeting quality $C_{sd}(t)$ is assigned a weight, such that as the entry becomes older, it contributes less to the overall forecasting. The base case value of recursion is $F_{sd}(0)$, given as:

$$F_{sd}(0) = \frac{1}{\omega}\sum_{t=1}^{\omega} C_{sd}(t) = F_{sd}(t-1) + \frac{C_{sd}(t) - C_{sd}(t-1)}{\omega}. \tag{4.5}$$

The above equation represents the moving average of the past meeting qualities of node *s* with *d* within interval $[1, \omega]$.

4.4. Simulation and Results

Simulations were performed using the Opportunistic Network Environment (ONE) simulator [4.13] by using synthetic mobility model and real connection traces [4.14]. For synthetic mobility, a large-scale DTN scenario is created using the map-based mobility model of the ONE simulator. A map of area 4,500m × 3,900m is marked with places, such as shops, homes, offices, meeting points, and bus stops. Independent groups of mobile nodes follow various routes on the roadmap. Table 4.2 presents the most common simulation settings. Moreover, some of the test runs indicate that the proposed *FAR* protocol exhibits the best performance for the values of $\phi = 0.6$ and $\omega = 50$.

Table 4.2. Simulation parameters.

Parameter	Value(s)
World size	4,500m × 4,000m
Simulation time per run	12 hours
Radio Interface	Speed:250Kbps (2Mbps), Range: 20 meters
Message	Size: 500KByte-1MByte, Interval: 1 per minute, TTL: 500 minutes
Buffer size	10-100MB (maximum buffer size that a node is willing to allocate for message distribution)
Nodes	Buses: 8, Cars: 20, Pedestrians: 72 Total: 100 (any node can be source as well as destination)
Node average Speed	Buses: 10-35 Km/h, Cars: 10-50 Km/h, Pedestrians: 0-5 Km/h
Mobility	City environment, Real connectivity traces

Furthermore, the following performance metrics were also considered.

$$Message\ Delivery\ Ratio = \frac{1}{M}\sum_{k=1}^{M} R_k\ . \tag{4.6}$$

In (4.6), the variable *M* is the total number of messages created, and $R_k = 1$, if message m_k is delivered; otherwise, $R_k = 0$.

$$Latency\ average = \frac{1}{\mathcal{M}}\sum_{k=1}^{\mathcal{M}}(Receive\ Time_k - Creation\ Time_k\ ), \tag{4.7}$$

where, $\mathcal{M}$ is the total number of messages received.

$$Overhead = \frac{Total\ msgs\ relayed - Total\ msgs\ delivered}{Total\ msgs\ delivered}\ . \tag{4.8}$$

To evaluate performance, the *FAR* protocol is compared with four state-of-the-art routing schemes namely: (i) *PRoPHET* [4.5], (ii) *Epidemic* [4.8], (iii) *Random* [4.8], and (iv) *Wave* [4.7].

Simulation results with synthetic mobility models are reported in Fig. 4.2(a)–Fig. 4.2(f) and Fig. 4.2(g)–Fig. 4.2(i) present the results with real-world connectivity traces.

Fig. 4.2(a)–Fig. 4.2(c) present the scalability performance of the *FAR* protocol, as the number of the nodes are increased within the system. Here, the *FAR* protocol outperforms all of the other routing schemes in terms of the delivery ratio, latency, and overhead. This is because the proposed scheme accurately forecasts the future contact durations by performing the online analysis of time-series data of previous contact durations. On the contrary, the *PRoPHET* protocol performs the future contact estimation on the basis of number of contacts without considering the duration of each of the contact. If the contact duration is not considered, then a message may be forwarded to a node that stays in contact with the destination for a very brief duration that is not enough of time to transfer messages.

The *Epidemic* protocol maximizes flooding to improve the message delivery, such that a node sends a message to all of the connected neighbors. However, higher flooding causes increased overhead. Flooding also increases the message drop rate in the networks with limited buffer spaces. Alternatively, the *FAR* protocol performs selective message replication that results in a decreased overhead (Fig. 4.2(c)) and higher delivery ratios (Fig. 4.2(a)). The *Random* protocol forwards a single message copy to any randomly selected neighbor. On the other hand, the *Wave* protocol maintains tracking lists to control flooding, such that a message relayed by a node a few moments earlier will not be relayed by the same node for a specific interval of time. Despite that, both the *Random* and the *Wave* protocols are resource conservative, and exhibit lower performance than the *FAR* protocol, as shown in Fig. 4.2. This is due to the fact that both of the protocols do not utilize the past meeting patterns of the nodes to perform message route estimations.

Simulations are also performed by increasing the transmission range of the nodes from 20m to 100m. By increasing the transmission range, the nodes have more time to stay connected and exchange messages. However, even with the increased transmission range, none of the algorithms outperform the *FAR* protocol (because of the accuracy of the prediction function utilized).

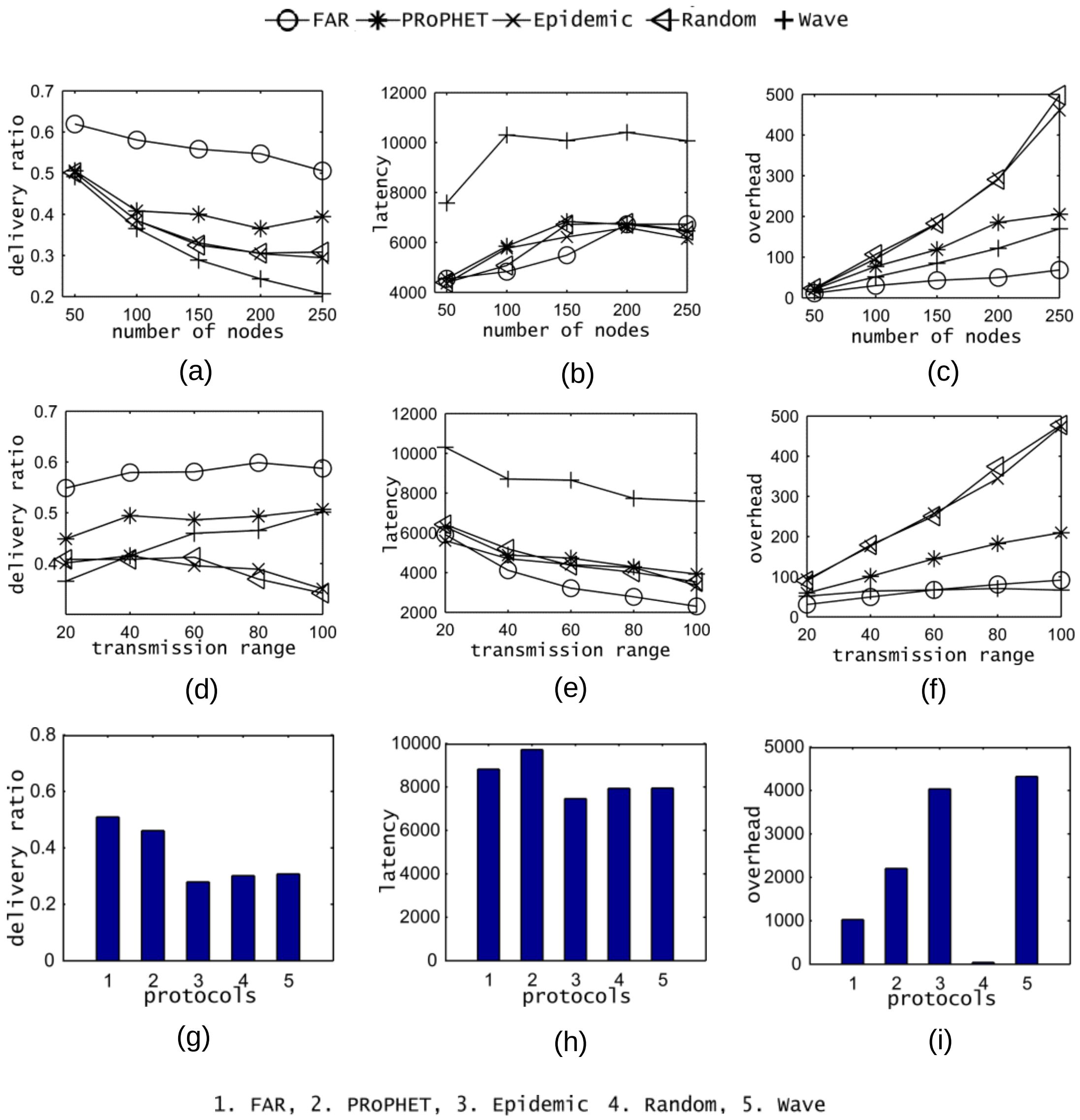


Fig. 4.2. Performance comparisons with synthetic mobility (a)–(f) and real connectivity trace (g)–(i).

The simulation results with real connectivity traces indicate that the *FAR* protocol performs better in terms of delivery ratios and overheads, as reported in Fig. 4.2(g)–Fig. 4.2(i). The overhead of the *Random* protocol is the minimum, as the *Random* protocol is a single-message copy forwarding scheme that has fewer transmissions per message. The *FAR* protocol takes advantage of the repetitive mobility patterns of the conference participants to achieve good performance. The latency metric of the *Epidemic*, *Random*, and *Wave* protocols is better than the *FAR* protocol (see Fig. 4.2(h)). However, the aforementioned phenomenon is at the expense of a low delivery ratio. Moreover, the *FAR* protocol exhibits the minimum overhead, as compared to the *PRoPHET* and *Epidemic* protocols, despite that the *FAR* protocol is a multiple copy replication scheme.

4.5. References

[4.1] B. Jedari and F. Xia, "A Survey on Routing and Data Dissemination in Opportunistic Mobile Social Networks," IEEE Communications Surveys and Tutorials, DOI: 10.1109/SURV.2014. 022714.00153, 2014

[4.2] M. Conti and M. Kumar, "Opportunities in Opportunistic Computing," *IEEE Computer*, vol. 43, no. 1, pp. 42-50, 2010.

[4.3] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," *IETF. RFC 3561*, March, 2014.

[4.4] C. P. Mayer, "Hybrid Routing in Delay Tolerant Networks," *KIT Scientific Publishing*, 2012.

[4.5] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "Probabilistic Routing Protocol for Intermittently Connected Networks," http://tools.ietf.org/html/rfc6693, 2012, accessed on Feb 2014.

[4.6] Z. Guo, B. Wang, and J. Cui, "Generic prediction assisted single-copy routing in underwater delay tolerant sensor networks," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1136–1149, 2013.

[4.7] J. Ott, A. Keränen, , and E. Hyytiä, "BeachNet: Propagation-based Info-Sharing in Mostly Static Networks," In *Proc. ExtremeCom.*, 2011.

[4.8] A. Balasubramanian, B. Levine, and A. Venkataramani, "Replication Routing in DTNs: A Resource Allocation Approach," *IEEE Trans. Netw.*, vol. 18, no. 2, pp.596-609, 2010.

[4.9] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *Technical Report CS-200006, Duke University*.

[4.10] T. Spyropoulos, T. Turletti, and K. Obraczka, "Routing in Delay-Tolerant Networks Comprising Heterogeneous Node Populations," *IEEE transactions on mobile computing*, vol. 8, no. 8, pp. 1132-1147, 2009.

[4.11] I. Rhee, M. Shin, S. Hong, K. Lee, S.J. Kim, and S. Chong, "On the Levy-Walk Nature of Human Mobility Networking," *IEEE/ACM Trans on Networking*, vol. 19, no. 3, pp. 630-643, 2011.

[4.12] C. Song, Z. Qu, N. Blumm, and A. Barabási, "Limits of Predictability in Human Mobility," *Sci. Mag*, vol. 327, no. 5968, pp.1018-1021.

[4.13] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE simulator for DTN protocol evaluation," *2nd Intl. Conf. Sim. Tools. Tech.* 2009.

[4.14] Infocom06 connectivity traces on "CRAWDAD website," http://crawdad.cs.dartmouth.edu, accessed on 05 March, 2014.

5. A CHECKPOINT BASED MESSAGE FORWARDING SCHEME FOR OPPORTUNISTIC COMMUNICATION

This paper was presented in *26th European Conference on Modeling and Simulation (ECMS)*, Koblenz, Germany, May 2012. The authors of this paper is Osman Khalid, Samee U. Khan, Joanna Kolodziej, Limin Zhang, Juan Li, Khizar Hayat, Sajjad A. Madani, Lizhe Wang, and Dan Chen.

5.1. Contributions

This chapter presents a concept of Checkpoint (CP) based message forwarding in opportunistic mobile networks [5.1], [5.2]. The CPs are autonomous high-end wireless devices with large buffer storage, and are responsible for temporarily storing the messages to be forwarded. Apart from having own memory and processing capabilities, each CP is a node that represents a specific region on the city map and maintains the information of geographic locations of all the other CPs. The CPs are deployed at various places within the city parameter that are covered by bus routes and where human meeting frequencies are higher. In contrast to the existing work, the CPs are not communication dependent on any fixed backbone network, and can be easily relocated. For the simulative analysis, a synthetic human mobility model in ONE simulator is designed for the city of Fargo, ND, USA. The model is tested over various opportunistic routing protocols and the results indicate that using CP overlay over the existing opportunistic communication significantly decreases message delivery time as well as buffer usage.

The rest of the chapter is organized as follows. First, related work is described with a comparison to the CP approach. Next, the CP architecture is discussed along with simulation scenario. Finally, the simulation results are discussed with conclusion and future work.

5.2. Checkpoint Architecture

The major components of the CP architecture include static CP nodes, mobile wireless nodes, buses, and additional components (e.g. Internet connected Access Points) that may be integrated with CP architecture.

5.2.1. CP Nodes

A CP is a wireless node that can be any low cost custom design hardware. The basic components of a CP node are but not limited to processor, memory, solar powered batteries, multiple interfaces (Standard Ethernet, 802.11b/g/n, Bluetooth), GPS, and storage. Each CP may optionally have the record of GPS coordinates of other CPs in the area. The coordinates are relayed through mobile nodes to the neighboring CPs along with normal data packets. Moreover, we assume that the CPs can be reconfigured with various opportunistic communications' routing protocols. For example, if routing protocol used by CP is the *PRoPHET* protocol [5.3], then the CP maintains a database of recent encounters with mobile nodes for certain amount of time T after which the old data is overwritten to make space for new records. In addition to message routing, other tasks that may be assigned to CP include content distribution at specific intervals, such as travel information, promos, news and information caching. The main fields maintained in the database of each $Checkpoint_i$ are indicated in Table 5.1.

Table 5.1. Database fields for a *$Checkpoint_i$*.

Node ID	The identification of a mobile node N_i
Number of contacts	The number of contacts a node made with $Checkpoint_i$
CP ID	CP with which the node made contacts. Here multiple entries are possible, as a node may make contacts with more than one CP.
Coordinates	The GPS coordinates of a CP last contacted by a node. This helps in synchronization of location information of CPs throughout the region.
Last Contact Time	Time of last contact with $Checkpoint_i$
Expected Contact Time	This time is predicted based on the node's history of contacts.

5.2.2. Mobile Nodes

The mobile nodes are pedestrians, cars, and buses with each node carrying 802.11b/g/n enabled wireless sets. This assumption makes sense due to a market research report [5.4] according to which in year 2009 alone, a total of 144 million mobile phones were shipped with Wi-Fi capability and it is further estimated that such phones may reach 66% of the total shipments till 2015 [5.5]. In the CP approach, whenever a mobile node interacts with a CP, the mobile node shares database with the CP by sending a light-weight summary vector and then the CP updates own database with new information about the node. Any two mobile nodes on encounter, exchange messages for storing, carrying, and forwarding, as well as the metadata of each node's visits to particular CPs. The minimum data structure required at each mobile node is reflected in Table 5.2.

Table 5.2. Database fields for a $MobileNode_i$.

CPs	The IDs and coordinates of every CP a node has visited in past.
Number of Contacts	The number of contacts a node has made with each $Checkpoint_i$
Last Contact Time	Time of last contact with $Checkpoint_i$

5.2.3. Buses Nodes

Buses are the message carriers or relay nodes in CP architecture. Each bus follows a fixed route and schedule and may pass through more than one CP (bus stop) on predefined timings. Moreover, after every scheduled round, every bus returns to a central bus station. The buses may be installed with any custom made wireless hardware having communication and storage ability to store the received messages to be delivered to the destination CPs.

5.3. Message Format

The minimum fields a message may have are source address, destination address, destination CP address, and payload. To find the destination CP address, an online Google based custom map for CPs may be consulted that indicates the specific CPs deployed near a particular geographic location. The aforementioned map can be constructed temporally with the passage of time if the CPs relay their GPS coordinates along with the actual message.

5.4. Message Routing

The message forwarding in the presented approach depends on the opportunistic routing protocol the CP is configured with. If a CP utilizes encounter based routing, then source node uses mobility pattern and schedules of buses, to forward the packet to a CP that is located closer to the destination. As shown in Fig. 5.1, there are four regions A, B, C, and X, each covered by a CP. Buses relay messages between any two regions and each bus also visits a central bus station denoted by X. The time of arrival and departure of buses is predefined. If a source node is within the communication range of CP, the node forwards a single copy of message to CP, to be relayed by bus nodes.

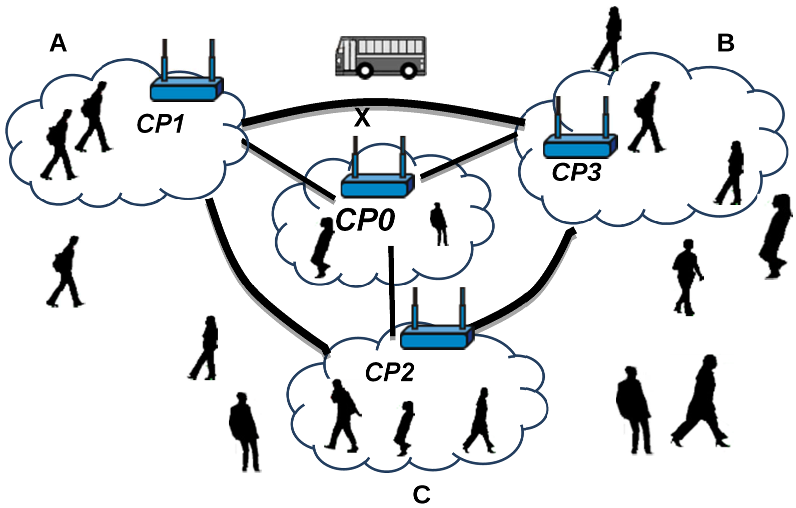


Fig. 5.1. CP architecture with CPs connected through bus nodes.

However, if the source node is outside the communication range of CP, then the source node opportunistically forwards the packets to the neighboring nodes in an attempt to allow at least one copy of message to reach the nearest CP or to the final destination. The aforementioned message forwarding can be accomplished by utilizing any resource efficient replication based opportunistic/DTN protocol (e.g. *Spray and Wait* [5.6]). The message copy is stored in a CP until the message is relayed to the next mobile node such as pedestrian, car, or bus. As the CPs are deployed on places where human mobility is higher, this increases the chance of message delivery to the destination. However, if CPs are deployed randomly (e.g. in case of post disaster scenario), then the GPS coordinates may be utilized in the calculation of minimum length routes among the source and destination CPs. There might be the case that a packet's destination information is not present in a CPs database. In that case, the destination is searched on the other CPs until the message TTL expires or all the CPs are searched.

5.5. Simulation

The simulation tool selected for the evaluation of the proposed CP model is Opportunistic Network Environment (ONE) simulator [5.7] that has rich features available for simulating opportunistic networks with numerous mobility models. The map of Fargo city is exported from www.openstreetmap.org. Using open source GIS tool OPENJUMP [5.8], the map is post-processed and marked with various locations such as shops, homes, offices, meeting points, NDSU, and GTC bus station. Mobile nodes are divided into several groups and assigned to various locations on the map.

5.5.1. Scenario

For simulation, an area of 4×3 KM of the City of Fargo, ND, USA is selected as indicated in Fig. 5.2. The buses are tagged with route numbers and follow various schedules available on the Metro Area Transit website [5.9]. Each bus route has stops at various locations. At some of the stops the CPs are deployed, each identified with an ID and representing a geographic location within 100 meters radius. *CP5* is located in the central bus station where each route bus arrives or departs from. Two main shopping locations (West Acres and Walmart) are covered by *CP9* and *CP10* through which route 15 bus runs. Moreover, *CP1*, *CP2*, and *CP3* are installed at junctions that are not covered by any bus route and the messages are relayed among the CPs with the help of public automobiles. The human mobile nodes are Wi-Fi/Bluetooth devices that are distributed throughout the simulation area. As an example of the current scenario, if *CP4* receives a message to be relayed, *CP4* stores the message and waits for route 13 bus to arrive. On arrival, bus 13 relays the message to the central bus station where the message is received by *CP5*. *CP5* checks the message destination in database and routes the message to the final destination's CP. If no such entry is found, the message is routed to all the

CPs after setting a TTL, on expiry of which, the message is deleted if not delivered (depending on the routing protocol).

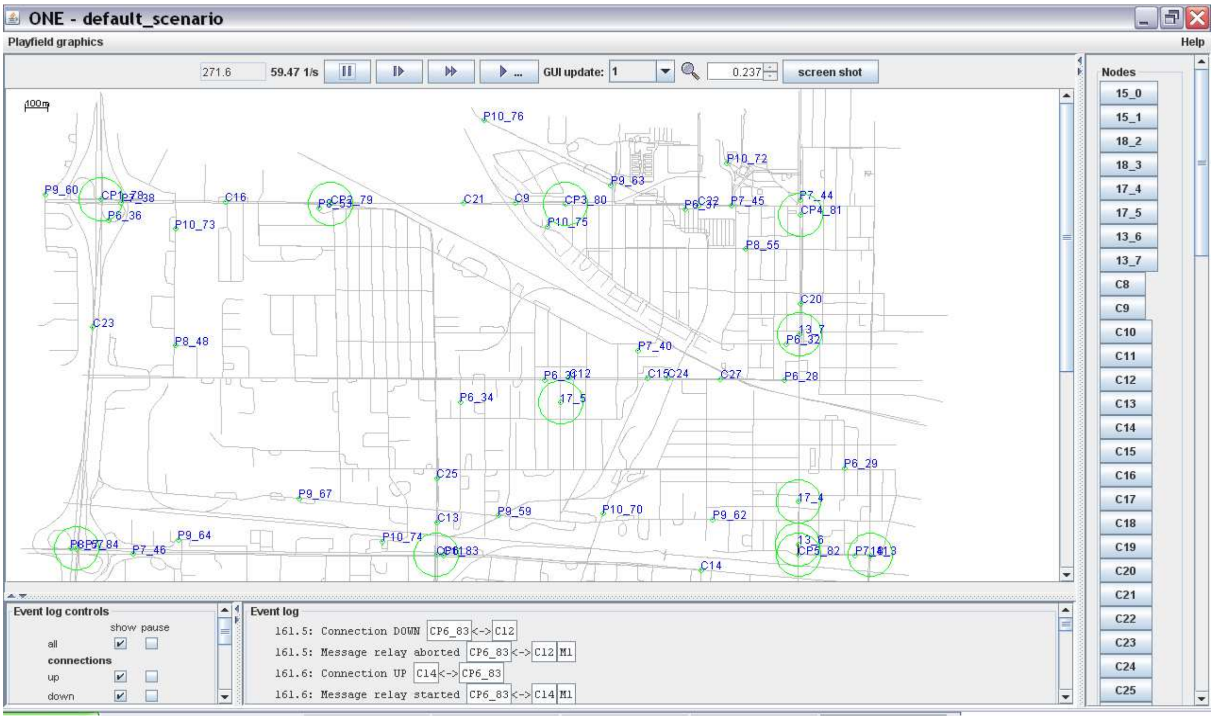


Fig. 5.2. Checkpoint based simulation model in ONE, with circles representing the deployed CPs.

5.5.2. Simulation Parameters

For the simulation, various combinations of parameters with range of values are selected. The simulation world is Fargo city map. Table 5.3 indicates the selected simulation parameters.

Table 5.3. Simulation parameters used in ONE.

Parameter	Value
World size	4250 × 3900 m
Simulation time per run	43200s == 12h
Bluetooth Interface transmit rate	250kbps
Bluetooth interface transmit range	10 m
High speed interface type	Broadcast Interface
High speed interface transmit speed	10Mbps
High speed interface range	100 m
Total number of node groups in the scenario	21
Nodes mobility model	Map based movement
Car / pedestrians nodes buffer size	250Mb
Car / pedestrians wait times	0, 120 s
Car / pedestrians speed range	0.5 to 1.5 m/s
Car / pedestrian node interface	Bluetooth
Message TTL	300 min
Bus nodes buffer size	500Mb
Bus nodes wait time	10, 30 s
Bus nodes speed	7, 10 m/s
Bus nodes interfaces	Bluetooth and High speed
Cars nodes speed range	2.7, 13.9 m/s
Total Checkpoints	11
Checkpoints buffer size	500Mb
Checkpoints interfaces	Bluetooth and High speed
Events interval	45, 55 s
Message size range	500KB – 1MB
Total message generating nodes	78
Warm up period	1000 s

5.6. Results

To perform model evaluation, a series of simulations have been performed in ONE Simulator. A single simulation time is 12 hours (43200s). The warm-up period is 1000s that is required for each CP and mobile node to have sufficient information in database. The CP architecture is evaluated for: **(a)** message delivery ratio, **(b)** buffer utilization, and **(c)** average message delay. The number of messages, CPs, buses, mobility pattern, nodes speed, transmission range, number of messages, and buffer sizes are altered in each simulation run to analyze the effect on average message delivery ratio, buffer utilization, and delivery delay.

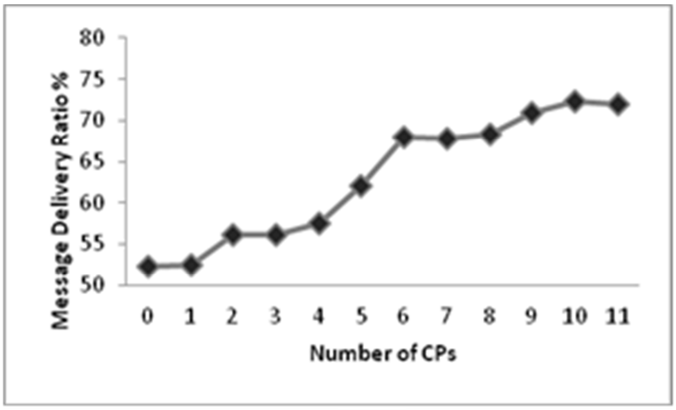
5.6.1. Effect of CP Deployment on Average Delay and Message Delivery Ratio

One of the most important and challenging task is the selection of ideal places for the deployment of CPs. Several factors influence the selection of an ideal location [5.10], [5.11] and the most important is the human meetings frequency at a particular place. To examine the effect of CPs deployment, *CP1*, *CP7*, and *CP11* are deployed at locations where meeting frequencies are lesser, as compared to *CP4*, *CP5*, *CP6*, *CP9,* and *CP10* that are located at shopping malls, bus stations, and North Dakota State University (NDSU). The simulation is run multiple times to observe the effect of CP deployment on message delivery ratio. The results in Fig. 5.3 indicate that for *CP1*, *CP7*, and *CP11* the message delivery ratio is lesser as compared to *CP5*, *CP6*, and *CP9* that are deployed considering the higher probability of meetings at these places. Therefore, the CPs would have more desirable outcomes if human mobility and meeting schedules are exploited before deployment.

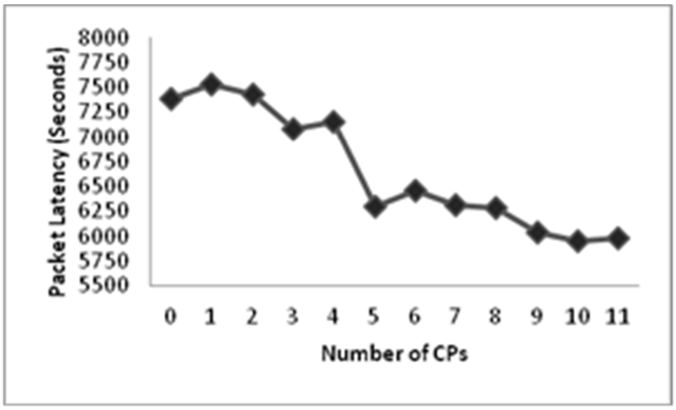
Fig. 5.3. Effect of CP deployment on message delivery ratio.

5.6.2. Effects of the Number of CPs on Message Delivery Ratio and Average Delay

The effect of the number of CPs is observed on message delivery ratio and average delay in Fig. 5.4. The test is run by increasing number of CPs while keeping number of buses and mobile nodes constant.



(a)



(b)

Fig. 5.4. Effect of CPs on (a) message delivery ratio and (b) packet latency.

From Fig. 5.4(a) we can see that message delivery ratio improves by increasing number of checkpoints in the area. In Fig. 5.4(b) we can observe that there is no significant decrease in packet latency until CP 5 is deployed. When CP 5 is deployed on bus station, there is remarkable decrease in packet latency as all the buses visit the common place. Therefore, due to the increase in human meetings at a common point, the packet latency also decreases.

5.6.3. CP Model Evaluation with Human Mobility Pattern

A place, where human meeting frequency is higher and repetitive, will be having higher message delivery probability. In such case DTN routing protocol such as *PRoPHET* would be more effective, as compared to the other routing protocols (e.g. *Spray and Wait* [5.6] and *Epidemic* [5.12]) that do not consider the node encounters pattern in making routing decisions. To evaluate the effect of human mobility on DTN routing, we identified some areas on the map as meeting points, where people tend to visit more frequently, and deployed a few checkpoints on these locations. Examples of such areas are shopping malls and main bus station GTC. We compared the performance of the three aforementioned routing protocols in terms of message delivery ratio and packet latency to investigate the effect of human mobility on these protocols. The simulation is run with same number of nodes for all three protocols and the results are indicated in Table 5.4.

Table 5.4. Performance of DTN protocols for human mobility.

	PRoPHET	Spray and Wait	Epidemic
Delivery Ratio %	71.90	62.39	73.51
Overhead ratio	27.64	7.52	34.77
Latency Avg. (s)	5964.51	5609.17	6163.54
Buffer time Avg. (s)	10249.51	16347.72	10225.18

In Table 5.4, latency avg. is average message delay from creation to delivery, overhead ratio is assessment of bandwidth efficiency, and buffer time avg. is average time the messages stayed in the buffer at each node. It can be observed that *Epidemic* routing has slightly higher delivery ratio as compared to *PRoPHET.* This is due to the fact that in *Epidemic* routing, the network is flooded with message copies such that each node replicates the message received and forwards to the next node. Therefore, the chances for message to reach the final destination also increase. However, from Table 5.4 we can see that message flooding increases bandwidth overhead ratio, and latency in *Epidemic* routing as compared to *PRoPHET* and *Spray and Wait* routing protocols. The buffer utilization time of *PRoPHET* is little higher than *Epidemic* as the message has to wait on a checkpoint before being delivered to the next mobile node (e.g. bus and car) towards the destination. The message delivery ratio is minimum for *Spray and Wait* routing due to the limit on number of message copies in the network and that leads to lower bandwidth overhead. Therefore, from Table 5.4 we can conclude that if human meeting schedules are exploited in message forwarding, then *PRoPHET* routing protocol surpasses the other two protocols in better performance.

5.6.4. CP Model Evaluation with RWP Mobility Pattern

To further examine the effect of mobility on DTN routing protocols, simulation is performed with a non-restricted Random Way Point (RWP) mobility pattern. Such mobility pattern may be observed in post disaster scenarios, where people are moving from one relief camp to another and then back to disaster locations not following a specific mobility pattern. The evaluation of the three DTN protocols is performed by randomly placing 11 CPs with fixed number of mobile nodes having various speeds. Simulation is run to study the effect of random mobility on message latency, buffer utilization, and message delivery ratio. Table 5.5 shows the effect of random waypoint mobility.

Table 5.5. Performance of DTN protocols for RWP mobility.

	PRoPHET	Spray and Wait	Epidemic
Delivery Ratio %	6.53	6.19	7.51
Overhead Ratio	37.42	22.05	44.68
Latency Avg. (s)	8705.82	8426.67	8661.85
Buffer time Avg. (s)	11451.37	14184.71	13090.77

From Table 5.5, we can observe that in random waypoint mobility, the message delivery ratio is significantly dropped. This is due to the fact that in random way point mobility, the frequency of nodes travel is higher towards the center of the map as compared to the map boundaries. The aforementioned fact can be further verified by looking at Fig. 5.5, which indicates that the message delivery ratio is higher at CP6 and C8, in all three protocols. Because, both the checkpoints (CP6 and CP8) are placed more closed towards the center of the map.

It can be further observed from Table 5.5 that there is no significant difference in the performance of the three protocols, as the *PRoPHET* cannot make use of human mobility pattern to forward messages towards the frequently visited points.

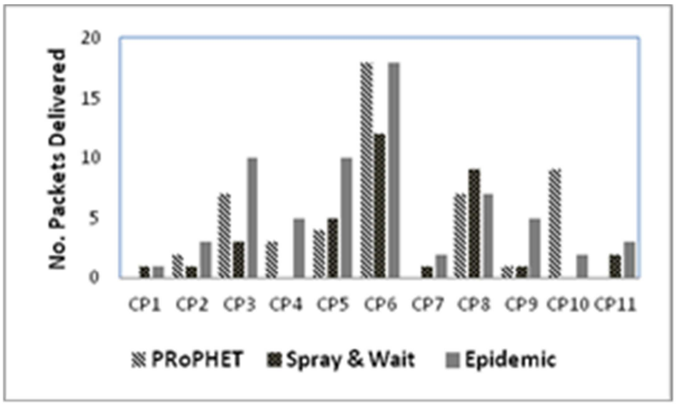


Fig. 5.5. CP6 is located near center of map where nodes mobility is maximum.

5.7. References

[5.1] M. Conti and M. Kumar, “Opportunities in Opportunistic Computing,” *IEEE Computer,* vol. 43, no. 1, pp. 42-50, 2010.

[5.2] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz, “Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges,” *IEEE Communications Surveys and Tutorials,* vol. 14, no. 2, pp. 607-640, 2012.

[5.3] A. Lindgren, A. Doria, E. Davies, and S. Grasic, “Probabilistic Routing Protocol for Intermittently Connected Networks,” http://tools.ietf.org/html/rfc6693, 2012, accessed on Feb 2014.

[5.4] A. Ionut, “Wi-fi enabled mobile phone handsets in the US, 2010-2015,” Technical report. Market research analysis, Coda Research Consultancy, 2010.

[5.5] A. Ionut, “Wi-fi capable handsets: Residential and enterprise markets for wi-fi and dual-mode handsets,” Technical report, ABI Research, 2009.

[5.6] T. Spyropoulos, T. Turletti, and K. Obraczka, “Routing in Delay-Tolerant Networks Comprising Heterogeneous Node Populations,” IEEE transactions on mobile computing, vol. 8, no. 8, pp. 1132-1147, 2009.

[5.7] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE Simulator for DTN Protocol Evaluation.” In *Proceedings of SIMUTools, 2009, Rome, Italy*.

[5.8] OPENJUMP GIS Software, [online] http://www.openjump.org/, accessed on March, 2014.

[5.9] Fargo Moorhead Metro Area Transit, “www.matbus.com,” accessed on march, 2014

[5.10] S.U. Khan, T. Loukopoulos and H. Li, “Advances in Wireless, Mobile and P2P based Internet Protocols, Applications, and Architectures,” International Journal of Internet Protocol Technology. 6, No.1-2, pp. 1-2, 2011.

[5.11] S.U. Khan, “Approximate Optimal Sensor Placements in Grid Sensor Fields,” In Proceedings of 65th Semi-annual IEEE Vehicular Technology Conference (VTC), Dublin, Ireland, pp. 248-251, 2007.

[5.12] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” Technical Report CS-200006, Duke University.

6. OPPORTUNISTIC DATABANK: A CONTEXT AWARE ON-THE-FLY DATA CENTER FOR MOBILE NETWORKS

This chapter is published in “Handbook on Data Centers, S. U. Khan and A. Y. Zomaya, Eds., Springer-Verlag, New York, USA”. The authors of this paper are Osman Khalid, Samee U. Khan, Sajjad A. Madani, Khizar Hayat, Lizhe Wang, Dan Chan, and Rajiv Ranjan,

6.1. Contributions

In this chapter, our primary focus is to address the data replication problem in mobile networks. In wireless environments, nodes are able to opportunistically share their buffer storage [6.1]. Such sharing of storage resources allows the formation of on-the-fly data centers [6.2]. A widely recognized way of balancing the demands of storage space with bandwidth and battery life is through data replication [6.3], [6.4]. To decrease communication cost, and increase data accessibility, replication distributes additional copies of primary data items within the network. Despite various natures of mobile networks, such as mostly disconnected delay tolerant networks (DTNs), and partially connected mobile ad hoc networks (MANETs), the data replication strategies in both the types of networks share some commonalities. However, data replication is more challenging in DTNs because of the absence of end-to-end connectivity paths between source and destination nodes. In Chapter 2, we presented a brief overview of some of the well-known strategies for replica placement in MANETs and DTNs.

We present a cost effective data replication scheme that relies on the collaboration of the nodes within the network to make replica placement decisions. The proposed scheme intends to perform replica placement in a way that not only restricts the number of replicas in the network, but also improves the average delivery probability. We compare our scheme with the selected replica placement approaches in DTNs, and test the proposed model on real-world and as well as synthetic trace datasets. The rest of the chapter is organized as follows. Section 6.2 presents the network model. Our proposed replica distribution scheme is presented in Section 6.3, and finally Section 6.4 presents empirical setup and results.

6.2. Network Model

We consider a DTN of a set of N mobile nodes. As indicated in Fig. 6.1, we divide the network nodes (represented by filled circles) into three types: **(a)** producers, **(b)** consumers, and **(c)** relays. Producers hold the original data item, which may be a measurement from a sensor network, or any piece of information generated by a node, such as emergency information, and weather information. Consumers are the information requesters that are the nodes that act as sink for the information item. A relay node holds replicas of data items on behalf of other nodes. The nodes are able to change their roles in our particular network scenario. Moreover, when two nodes make contact, both nodes make use of in-band control signaling to exchange their locally maintained network state information. In the following, we elaborate few assumptions we make for the proposed model.

Each mobile node has a unique network identifier. The producer nodes generate the data items, with each data item having a unique identity. A data item can have many replicas in the network. For simplicity, we assume the same meaning of ‘message’ and ‘replica’. A producer can directly serve the consumer node on making an opportunistic contact, or the replica can be relayed through relay nodes towards consumers. When two nodes make contact, the sender node may retain the replica, or delete it from local buffer, after transferring the replica to the neighbor. Such decision is based on probability measures that we illustrated in our model. Few relay nodes, such as buses, may follow scheduled mobility patterns, while others, such as pedestrians may follow scheduled, as well as random mobility models. The nodes share only a portion of their buffer capacity for the opportunistic data storage. Moreover, during the opportunistic contacts, the mobility of nodes restricts the amount of data transferred due to limited contact duration. Therefore, we formally state the replication problem in DTNs as *"Given a limited duration opportunistic contact between two nodes, what replicas must be selected for exchange between the nodes, so that they contribute in the global optimization of network overhead and message delivery percentage?"* The most frequent notations used in the chapter are shown in Table 6.1.

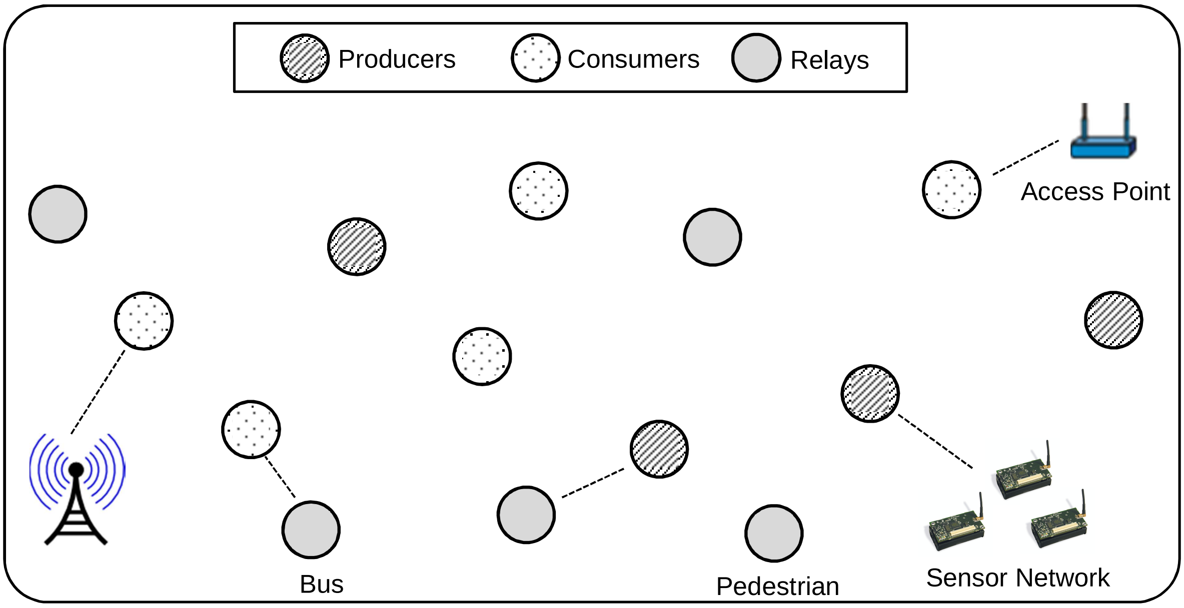


Fig. 6.1. An example of heterogeneous DTN network.

Table 6.1. Notations and their meanings.

Notation	Meaning
m^k	Message *k*
T_L^k	Life time of kth
T_w^k	Time the kth message spent waiting in buffers
T_t^k	Message transmission time
X^k	Random variable indicating the additional time that kth message might wait before delivery
τ	Time when a node makes contact with another node.
$Z_i^j(\tau)$	Mean inter-contact time between node *i* and *j* computed at time τ

As discussed earlier, in DTNs, nodes cannot maintain global network state. Therefore, we assume that nodes exchange network information during the opportunistic contacts. We denote the contact durations and inter-contact times between any two nodes i and j as C_i^j and I_i^j, respectively. Each node is maintaining a 2-tuple time-series information given as $< C_i^j[t]$, $I_i^j[t] >$, where $t = 1, 2, 3, \dots \omega$. The parameter ω is the index of last entry in the time-series data.

6.3. Hybrid Scheme for Message Replication (HSM) for DTN Environments

In this section, we present our scheme for message replication in DTNs. We call the scheme as hybrid, as we are also considering the occasional presence of MANET like environments in our network, when for example, the pedestrians stay closer to each other for longer durations, or the nodes are communicating with road side base stations.

Suppose a node i has a message m^kthat is requested by node d. At a time instant τ, the node i makes contact with a relay node j. At this occasion, the node i has to decide whether or not to replicate m^k on node j in a hope that node j might carry forward replica to node d. The node i will replicate m^kon node j, if and only if, node j's utility value is greater than node i's utility value for the replica m^k. The aforementioned utility value depends on: **(a)** the probability that a node will deliver message to destination before the life time expiry of message, and **(b)** the probability that the node will stay in contact with a message's destination for a duration greater than time required to transfer the message. If node j exhibits greater values of (a) and (b) as compared to the node i, the replica will be transferred to node j, and subsequently node i will delete the replica from local buffer. Otherwise, after transferring replica to node j, the node i will retain local copy of replica. The motivation behind such approach is to remove the excessive replicas from the network to conserve storage by placing replicas on nodes that appear to be more central in the network.

Let T_w^k be the time the message m^k has spent waiting in buffers since creation, and T_L^k be the life time of message m^k. We denote X^k to be a random variable representing the additional time that m^k might wait before reaching destination. Then, we define message's utility as the probability that the message will be delivered to the destination d before the life time expiry [6.5], given as $U^k = P[T_w^k + X^k < T_L^k]$. This can also be represented as:

$$U^k = P\left[X^k < T_L^k - T_w^k\right]. \tag{6.1}$$

In (6.1), we need to find the probability that additional wait time of replica is less than remaining life time. As the message is transferred only during an opportunistic contact, the probability in (6.1) is same as the probability that the node i will make a contact with node d, before the expiry of message. We call such probability as utility value of node i for the current message:

$$U_{i,d}^k = P[Z_i^d(\tau) < T_L^k - T_w^k]\,. \tag{6.2}$$

In the above equation, $Z_i^d(\tau)$ is mean inter-contact time between nodes i and d at time τ. The network nodes are cumulating their inter-contact time information in the form of bounded time-series data. Moreover, a few nodes (such as buses) are following partially scheduled mobility patterns. Therefore, we can apply exponential smoothing to forecast the value of inter-contact time between the node i and node d, as given below:

$$Z_i^d(\tau) = (1-\alpha)^{\tau-1} \cdot s[0] + \sum_{k=0}^{\tau-1} \alpha \cdot (1-\alpha)^{\tau-k-1} \cdot I_i^d[k] \tag{6.3}$$

In the above equation, the parameter $0 \le \alpha \le 1$ is time-series smoothing constant, $I_i^d[k]$ is inter-contact time of node i with d at time instant k, $s[\tau]$ is the base value of recursion, and $Z_i^d(\tau)$ is the forecasted inter-contact time node i with d. As the mobile devices are limited in memory and processing, we cannot store unlimited time-series data of the past meetings.

Therefore, we set a limit on the maximum number of entries stored per node in the form of a sliding time window $1 \leq t \leq \omega$, where the entry at $\tau = \omega$ represents the latest meeting. The more recent entries within the range $[1, \omega]$ must be given higher weightage than the others to ensure information freshness. Therefore, we assign progressively decreasing weights to the older entries, such that, as the entry becomes older it contributes less to the overall forecasting. The base case value of recursion $s[\tau]$ computed at time instant τ is given as:

$$s[0] = \frac{1}{n} \cdot \sum_{j=0}^{n-1} I_i^d[\tau - j] \tag{6.4}$$

The above equation is the simple moving average of latest n entries of the inter-contact times I_i^d between i and d. Let T_t^k be the time required to transfer a message m^k when two nodes make contact. Assuming that neighbour node has sufficient buffer space, the message will be successfully transferred to neighbour if and only if the contact duration of sender and receiving neighbour is greater than the required message transfer time T_t^k. Therefore, we compute the utility $V_{i,d}^k = P[T_t^k < C_i^d(\tau)]$, indicating the probability that the message will be transferred between node i and d in contact duration C_i^d. To compute the aforementioned probability, we need to find the estimated value of contact durations between nodes i and d. By replacing C_i^d with I_i^d in (6.3) and (6.4), we get the forecasted value of contact duration $C_i^d(\tau)$. Fig. 6.2 illustrates the flow of our replica placement scheme *HSM*.

As reflected from Fig. 6.2, the procedure attempts to remove the redundant copies of messages from nodes' buffers, and attempts to allocate replicas on more appropriate nodes in terms of contact durations and inter-contact times with the destinations.

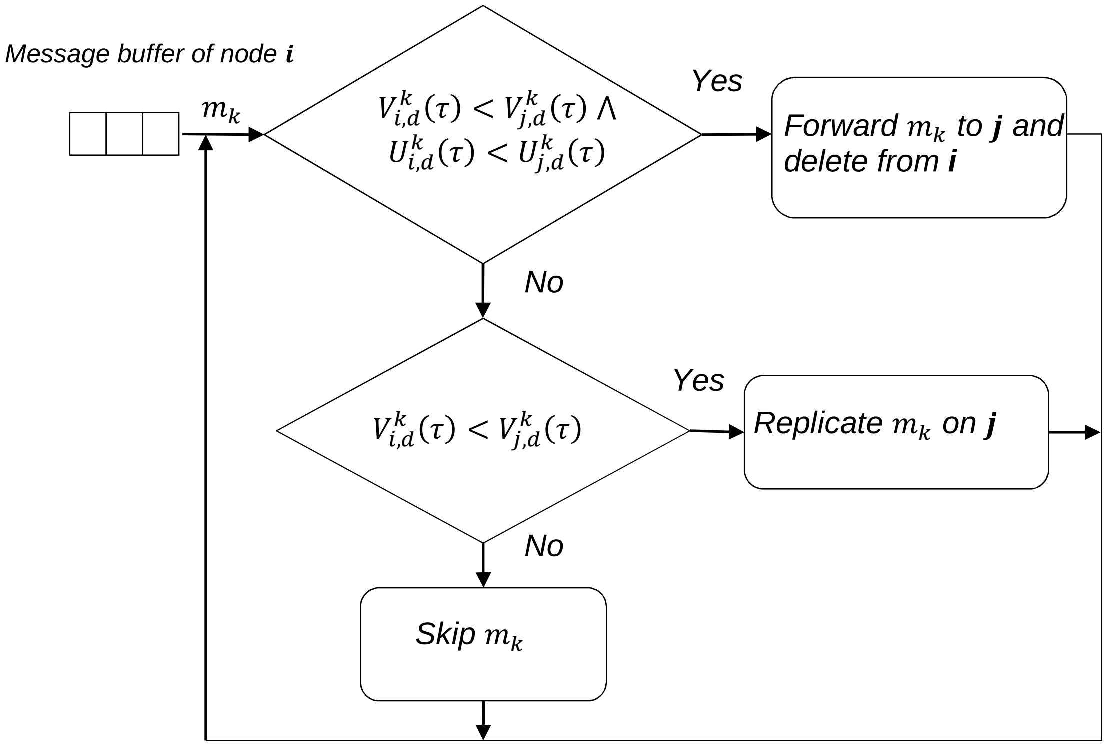


Fig. 6.2. Flowchart of proposed replication scheme.

6.4. Empirical Setups and Results

In this section we present the performance analysis of the presented replication scheme *HRM*. Simulations are performed with the Opportunistic Network Environment (ONE) simulator [6.6] by using synthetic mobility model as well as real connection traces of participants of the Infocom 2005 conference [6.7]. The synthetic mobility model consists of several independent groups of mobile nodes including pedestrians, buses, cars, and access points. Some mobile nodes, such as pedestrians, follow random mobility patterns, whereas buses follow scheduled mobility patterns. The car nodes follow paths representing roads on map. The parameters considered for simulations are: **(a)** nodes' range 50-100, **(b)** world size 4250×3900m, **(c)** time per simulation run 12h, **(d)** transmission range 20m, **(e)** message size 500KB-1MB, **(f)** message time to live (TTL) 500min, and **(g)** buffer size 10-100MB. The world size is taken large enough to ensure that nodes are far enough to represent a DTN environment. The *HRM* scheme gives best performance for values of $\alpha = 0.6$, $\omega = 50$, and n=10 determined empirically under numerous simulation runs. In the following subsection we discuss the performance metrics considered for evaluation.

6.4.1. Performance Metrics

To evaluate the performance of the presented scheme, we consider the following three performance metrics: **(a)** message delivery ratio, **(b)** latency, and **(c)** overhead.

- **Message delivery** ratio is the percentage of messages delivered successfully. The maximization of message delivery ratio is the major goal of any DTN replication scheme. Message delivery ratio is calculated as

$$Message\ Delivery\ Ratio = \frac{1}{M}\sum_{k=1}^{M} R_k\ . \tag{6.5}$$

In above equation, $R_k = 1$ if and only if message is delivered, otherwise $R_k = 0$.

- **Message latency** is the total time spent between message creation and delivery to the destination. The average latencies of messages contribute to the overall latency measure of replication scheme. A scheme must minimize latency but without compromising message delivery ratio. The latency (in seconds) is given by

$$Latency\ average = \frac{1}{N}\sum_{k=1}^{N} Receive\ Time_k - Creation\ Time_k\ . \tag{6.6}$$

- **Overhead** is the approximate measure of the consumption of bandwidth, energy, and storage by a replication scheme due to message transmissions. We calculate overhead as relative estimate of number of message transmissions:

$$Overhead = \frac{Total\ msgs\ relayed - Total\ msgs\ delivered}{Total\ msgs\ delivered}. \tag{6.7}$$

The overhead ratio indicates extra transmissions for each delivered message.

6.4.2. Related DTN Replication Schemes

To perform comparisons, we selected the following related replication schemes for DTNs: **(a)** *PRoPHET* [6.8], **(b)** *Epidemic* [6.9], **(c)** *Random* [6.5], and **(d)** *Wave* [6.10]. These schemes utilize various strategies and heuristics to replicate messages in the network. As the data routing in DTNs is based on data replication during opportunistic contacts, the aforementioned schemes can also be called as the routing schemes for DTNs.

6.4.3. Simulation Results

Simulation results with synthetic mobility model are indicated in Fig. 6.3(a)–Fig. 6.3(c), whereas Fig. 6.3(d)–Fig. 6.3(f), present the simulation results with real-world connectivity traces. As reflected in Fig. 6.3(a)–Fig. 6.3(c), the *HSM* scheme outperforms the rest of the replication schemes in terms of delivery ratio and overhead. This is because *HSM* accurately forecasts future contacts by performing online analysis of limited sized time-series data of previous contacts with varying qualities. On the contrary, the *PRoPHET* protocol performs future contact estimation on the basis of number of contacts without considering the time varying pattern of contact duration and inter-contact times.

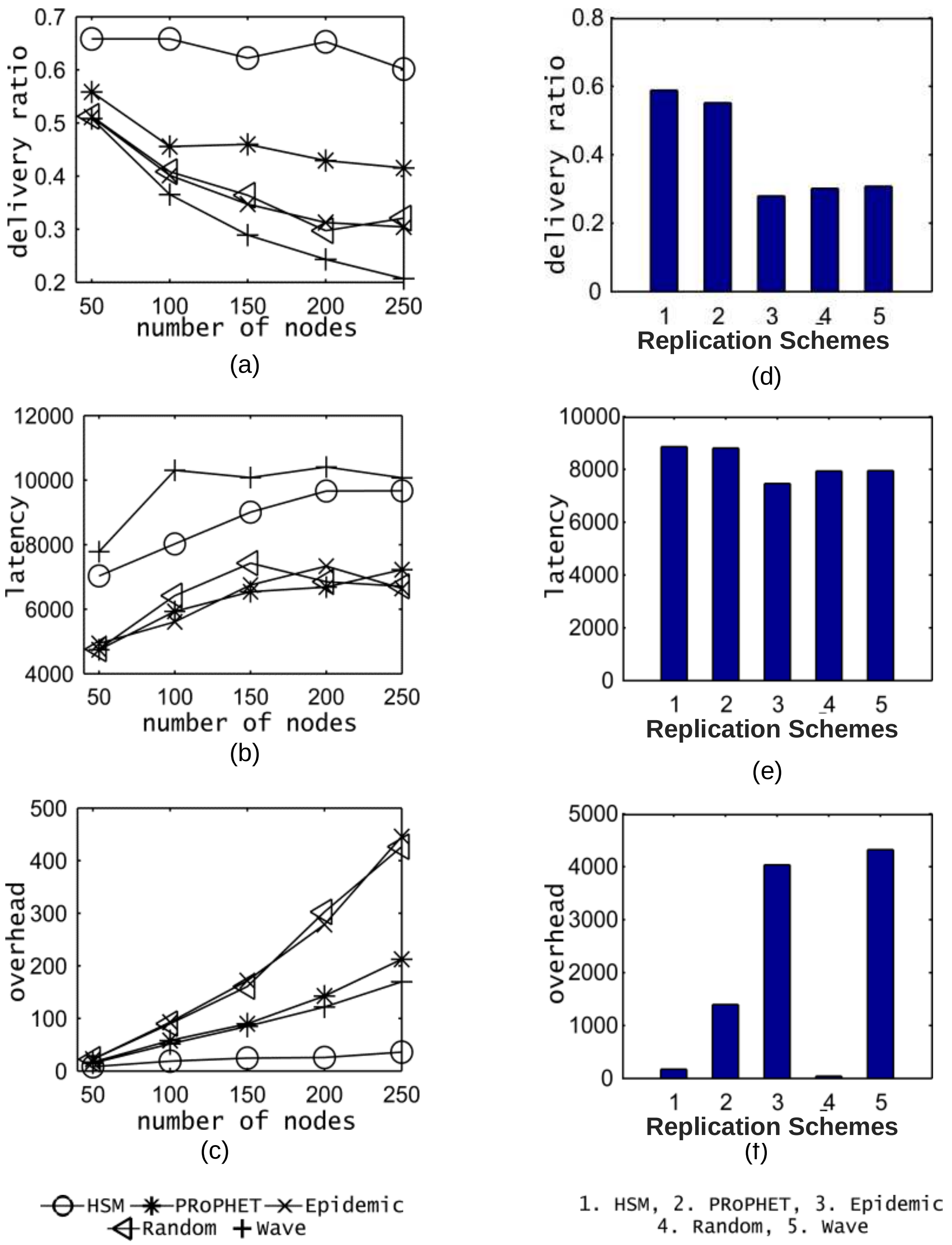


Fig. 6.3. Performance comparison results with synthetic mobility (a)–(c) and real mobility trace (d)–(f). The schemes compared are: 1. HSM, 2. PRoPHET, 3. Epidemic, 4. Random, and 5. Wave.

The *Epidemic* scheme maximizes the flooding to improve message delivery. However, higher flooding causes increased overhead and higher message drop rate in resource constrained network scenarios. Alternatively, *HSM* performs selective message replication resulting in decreased overhead (Fig. 6.3(c)) and higher delivery ratio (Fig. 6.3(a)). The *Random* scheme forwards single message copy to any randomly selected neighbor, whereas the *Wave* scheme performs replica flooding in a controlled manner. Despite that both the aforementioned schemes are resource conservative; they exhibit low performance than *HSM* as reflected in Fig. 6.3. This is because these schemes do not utilize the past meeting patterns to perform a node's utility estimations.

The simulation results with real-world connectivity traces indicated that *HSM* performed better for delivery ratio and overhead. As reflected in Fig. 6.3(d)–Fig. 6.3(f), the *HSM* scheme precisely utilized the meeting patterns of conference participants to perform future contact forecasts. The latency metric of *Epidemic*, *Random*, and *Wave* is better than *HSM* (Fig. 6.3(e)). However, this is at the expense of their low delivery ratio. Moreover, *HSM* exhibits minimum overhead as compared to *PRoPHET* and *Epidemic*, despite being multiple copy replication scheme.

6.5. References

[6.1] M. Conti and M. Kumar, "Opportunities in Opportunistic Computing," *IEEE Computer,* vol. 43, no. 1, pp. 42-50, 2010.

[6.2] Jorg Ott, Esa Hyyti, Pasi Lassila, Tobias Vaegs, and Jussi Kangasharju, "Floating content: Information sharing in urban areas," In *IEEE International Conference on Pervasive Computing and Communications (PerCom),* March, 2011, pp. 136-146.

[6.3] Dan Hirsch and Sanjay Madria, "Data Replication in Cooperative Mobile Ad-Hoc Networks," *Mobile Networks and Applications,* vol. 18, no. 2, pp. 237-252, 2013.

[6.4] S. U. Khan, “Mosaic-Net: A Game Theoretical Method for Selection and Allocation of Replicas in Ad Hoc Networks,” *Journal of Supercomputing*, vol. 55, no. 3, pp. 321-366, 2011.

[6.5] A. Balasubramanian, B. Levine, and A. Venkataramani, “Replication Routing in DTNs: A Resource Allocation Approach,” *IEEE/ACM Transactions on Networking*, vol.18, no.2, pp.596-609, 2010.

[6.6] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE simulator for DTN protocol evaluation,” in *Proc. of 2nd International Conference on Simulation Tools and Techniques*, 2009, doi.10.4108.

[6.7] Infocom06 connectivity traces on CRAWDAD website. [Online] http://crawdad.cs.dartmouth.edu/meta.php?name=cambridge/haggle#N100C4, 2014.

[6.8] A. Lindgren, A. Doria, A. Davies, and S. Grasic, “Probabilistic Routing Protocol for Intermittently Connected Networks,” http://tools.ietf.org/html/rfc6693, 2012, accessed on March, 2014.

[6.9] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” *Technical Report CS-200006, Duke University*.

[6.10] J. Ott, A. Keränen, and E. Hyytiä, “BeachNet: Propagation-based Info-Sharing in Mostly Static Networks,” In *Proceedings of ExtremeCom*, 2011.

7. APS: AN ADAPTIVE PROGNOSTIC MESSAGE ROUTING SCHEME FOR DELAY TOLERANT NETWORKS

This paper is submitted in *IEEE Communication Letters.* The authors of this paper are Osman Khalid, Nasir Ghani, and Samee U. Khan.

7.1. Overview

DTNs are characterized by frequent disruption/delays, and lack of end-to-end communication paths [7.1]. Applications of DTNs include disaster response systems, vehicular networks, wild life monitoring, and inter-planetary communication [7.2]. In such applications, the network stays disconnected most of the time, and there may not be contemporaneous end-to-end paths available between source and destination nodes. Therefore, the conventional routing protocols, such as AODV and DSDV [7.3] specifically designed for mobile ad hoc networks (MANETs) are inapplicable for DTN like scenarios. Message routing in DTNs is challenging due to the inherent uncertainty about network conditions that vary with time [7.4]. To exchange messages, nodes have to rely on opportunistic contacts during which a node decides whether or not to forward/replicate the message to the neighbor node. Such decisions are typically guided by the desire to control the number of message replicas in the network. There is no guarantee that a message eventually reaches the destination, as the message may be dropped due to network congestion, or life time expiry, yielding a best effort delivery service. Therefore, it is quite difficult for a DTN protocol to achieve a 100% message delivery.

7.1.1. Motivation

Several works, such as [7.4], [7.5], [7.6], [7.7] are based on message flooding where a node carrying a message replicates the message on every other node encountered. Flooding improves message delivery probability and reduces latency. However, with limited bandwidth and buffer space, flooding may result in an increase in congestion and message drop. To control the message flooding, DTN protocols, such as [7.8], set a limit on maximum number of replicas per message. In such protocols, the message is flooded to first *n* encountered nodes, after that *n* nodes can deliver the message only to the destination. By doing so, the protocols [7.3], [7.8] cause lesser overhead (number of extra transmissions per message). However, reduction in overhead is achieved at the expense of reduced delivery rate and increased latency. A few DTN protocols perform conditional replications such that a message is replicated on neighbor node when several conditions are met [7.1], [7.5], [7.6]. For instance, Lindgren *et al.* [7.6] proposed a probabilistic routing protocol that replicates a message on a neighbor node, if and only if, the neighbor has more frequently encountered with the message's destination. However, the protocol [7.6] imposes no limit on the maximum number of replicas per message and is not mobility cognizant. Single-copy forwarding schemes were proposed that relay a single copy of message towards the destination without making replicas [7.9]. Single-copy routing schemes are resource conservative as they utilize minimum buffer space and bandwidth. However, such schemes may also result in an increased delay and message drop, as the node having original message may never reach the destination.

7.1.2. Contributions

In light of the above discussion, there is still a pressing need to develop resource efficient routing protocols for DTNs that must also exhibit better message delivery rates. This chapter addresses the critical area of resource efficiency in DTNs and proposes a routing scheme *APS* that effectively restricts number of message replicas in the network to reduce the overhead. For many years, human mobility has been a focus of DTN research, and it is proven through numerous experiments that humans tend to follow repetitive schedule of meetings at similar places and times, and that the human mobility is predictable following a power law distribution [7.2], [7.10]. Song *et al.* [7.11] have shown through their experiments that human mobility is

93% predictable. Therefore, our proposed routing scheme *APS* learns from nodes' mobility patterns and temporal contacts to make precise predictions about future contacts. The *APS* protocol reduces the number of message replicas in the network by relaying messages to only those nodes that are more likely to make contact with messages' destinations. The proposed scheme utilizes time-series forecasting on nodes' contact patterns to determine optimal spatiotemporal routes for messages in a time varying topology. The *APS* protocol can be efficiently utilized in real-life applications to disseminate delay tolerant data, such as electronic newspapers, weather forecasts, movie trailers, and travel routes information in various parts of a city. Simulations conducted with two city-scale mobility scenarios indicated that the *APS* protocol shows better performance in terms of message delivery and overhead.

The rest of the chapter is organized as follows. The network model and assumptions are discussed in Section 7.2. In Section 7.3, we present the *APS* protocol design. The simulations and results are discussed in Section 7.4. Finally, Section 7.5 presents the conclusions.

7.2. Network Model and Assumptions

The network model we considered is a hybrid DTN consisting of mobile and static nodes [7.3]. Messages can only be transferred between two nodes when they are in each other's transmission range. During an opportunistic contact, a sender node may or may not replicate a message on a neighbour. A message may be directly delivered if the destination node is encountered, or relayed through intermediate nodes. Nodes have limited storage and message transfer opportunities. Moreover, a few of the mobile nodes, such as buses, follow schedules in mobility.

7.3. APS Protocol Design

We consider a DTN with N number of mobile nodes. When a node i makes a contact with a node j at time t, the node i records the contact details that are stored in the form of a finite time-series. Each data item of the time-series represents a contact quality denoted as $S_{ij}(t)$. The *APS* protocol quantifies the contact quality with total duration of contact, such that, the contacts of greater durations are assigned higher qualities. Specifically, greater the contact quality between any two nodes, the higher will be the message exchange probability. With the passage of time, each node *i* cumulates a bivariate time-series data consisting of contact time and contact quality for every other node in the network. To predict future contacts, the *APS* protocol utilizes the Auto Regressive Integrated Moving Average (ARIMA) model [7.12] on the time-series data stored by a node. The ARIMA model is widely applied for forecasting purposes and is generally represented as ARIMA(*p*,*d*,*q*). The three arguments *p, d,* and *q* provide the order of the three components of the model namely: **(a)** autoregressive, **(b)** integrated, and **(c)** moving average component. The ARIMA model is represented as:

$$\left(1 - \sum_{k=1}^{p} \phi_k B^k\right) \cdot Y_{ij}(t) = c + \left(1 + \sum_{k=1}^{q} \theta_k B^k\right) \cdot \varepsilon(t), \tag{7.1}$$

where $Y_{ij}(t) = (1 - B)^d S_{ij}(t)$, and the parameter *B* is the lag operator that shifts back the time-series data $S_{ij}(t)$ by one period, given as $B \cdot S_{ij}(t) = S_{ij}(t-1)$. The parameter $Y_{ij}(t)$ applies differencing of order *d* on the time-series to make the data stationary, whereas ϕ_k and θ_k are the parameters of autoregressive and moving average parts, respectively. The parameter $\varepsilon(t)$ are error terms assumed to have zero mean and *c* is a constant.

The left hand side of (7.1) represents the auto-regression model of order *p*, denoted as AR(*p*). The auto-regression model forecasts the time-series variable S using a linear combination of past values of the variable S, given as:

$$S_{ij}(t) = c + \phi_1 \cdot S_{ij}(t-1) + \phi_2 \cdot S_{ij}(t-2) + \cdots + \phi_p \cdot S_{ij}(t-p) + \varepsilon(t).$$

The right hand side of (7.1) represents the moving average part of the ARIMA model, denoted as MA(q). The moving average part utilizes past forecast errors to predict the time-series variable, represented as:

$$S_{ij}(t) = c + \varepsilon(t) + \theta_1 \cdot \varepsilon(t-1) + \theta_2 \cdot \varepsilon(t-2) + \cdots + \theta_q \cdot \varepsilon(t-q).$$

In the following subsections, we describe the various phases involved in forecasting the future contacts of nodes using ARIMA model.

7.3.1. Data Stationarity Tests

The ARIMA model requires that the time-series data must be stationary. A stationary time-series has statistical properties that do not vary over time. More precisely, the series must not be exhibiting any trends or seasonality, and must have constant mean and variance over time. If the time-series is not stationary, then the integrated term $(1-B)^d$ is introduced to apply *dth* order differencing on the data to make it stationary, where the *1st* order differencing is given by:

$$S_{ij}{}^1(t) = S_{ij}(t) - S_{ij}(t-1) = S_{ij}(t) - B \cdot S_{ij}(t) = (1-B)^1 \cdot S_{ij}(t).$$

To examine whether or not the nodes' temporal contacts in real DTN scenarios exhibit stationary behaviour, we performed analysis on real connectivity trace of UMass DieselNet that is an experimental bus based DTN [7.13]. The dataset consists of a series of contacts among buses at various time intervals. We randomly selected node pairs from the trace and utilized R statistical tool [7.14] to perform the data analysis. To check data stationarity, *unit root* tests are conducted on the time-series data of nodes' contact qualities. For that purpose, we utilized Augmented Dickey-Fuller (ADF) test available in the R's *forecast* package [7.14]. If the *p-value* of the ADF test is greater than 0.05, then the dataset is not stationary and differencing is required. The *p-value* we obtained on applying ADF test is 0.01 that indicates stationarity of the dataset. Fig. 7.1 shows the plot of nodes' contact qualities versus time.

The unit root tests are also performed on real connectivity trace of Infocom 2005 conference [7.15], and results indicated that the trace exhibits stationarity.

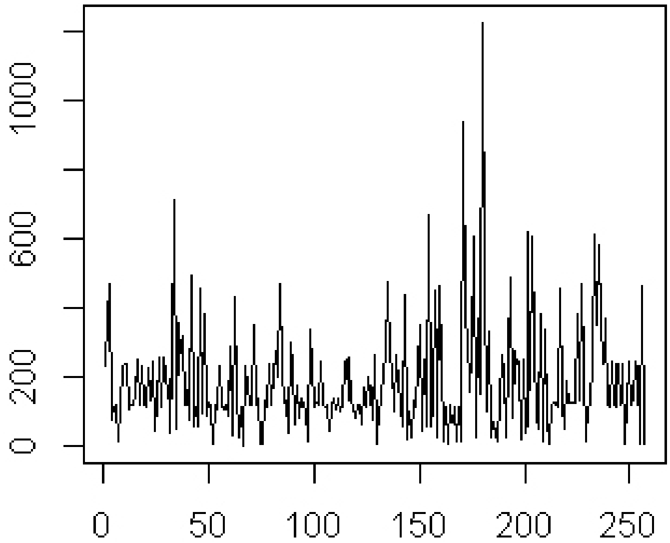


Fig. 7.1. Mobile carries are exchanging messages on making opportunistic contacts.

7.3.2. Computing p, d, and q

To compute the values of *p*, *d*, and *q* we applied *auto.arima* function [7.14] of R software on the time-series data. Moreover, as shown in Fig. 7.2, autocorrelation function (ACF) and partial autocorrelation functions (PACF) are also applied on the dataset that produced the order of model as ARIMA(2,0,2).

7.3.3. Forecasting

Suppose a source node *s* has a message for a destination node *d*. If node *s* makes a direct contact with node *d*, then the message is delivered to the destination. Otherwise, on making contact with a relay node *r* at time *t*, the node *s* decides whether or not to replicate the message on the relay *r* on the basis of the following ARIMA model equation.

$$S_{sd}(T+1) = \phi_1 \cdot S_{sd}(T) + \phi_2 \cdot S_{sd}(T-1) + \theta_1 \cdot \varepsilon(T) + \theta_2 \cdot \varepsilon(T-2). \qquad (7.2)$$

The above equation is obtained by substituting the values *p=2*, *d=0*, *q=2*, *c=0*, *t=T+1*, and $\varepsilon(T+1)=0$ in (7.1), where *s* and *d* represent the nodes *i* and *j*, respectively. The values of the parameters ϕ_k and θ_k are determined using maximum likelihood estimation [7.12]. According to (7.2), the node *s* replicates the message on relay *r*, if and only if, the relay *r* has a better forecast of future contact with the destination *d* that is represented as $S_{rd}(T+1) >$ $S_{sd}(T+1)$.

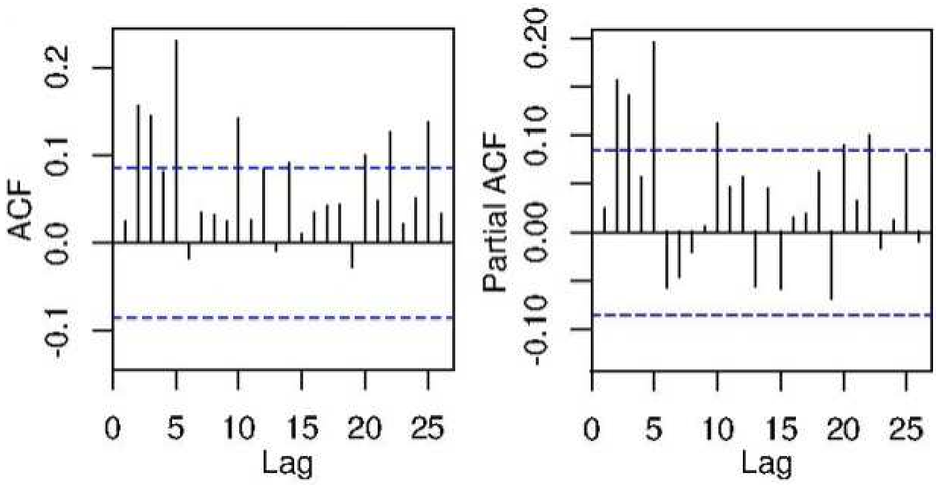


Fig. 7.2. ACF and PCF plots of nodes' contact durations.

7.4. Simulation Results and Discussions

To evaluate the performance, the *APS* protocol is implemented in Java. Two large-scale DTN scenarios are developed in the ONE simulator [7.16]. The Scenario-1 consists of a map of area 4,500 m × 4,000 m, whereas Scenario-2 constitutes a map of dimensions of 8,200m × 7,000m. Both of the scenarios consist of independent groups of mobile and static nodes, such as buses, cars, pedestrians, and access points. Buses run on specific routes following their respective schedules. Table 7.1 presents some of the common simulation settings used by the DTN scenarios. The simulation parameters are selected to closely depict the real-world scenarios.

Table 7.1. Simulation Settings.

Parameter	Value(s)
World size	Scenario 1: 4,500 meters × 4,000 meters, Scenario 2: 8,200 meters × 7,000 meters
Simulation time per run	12 hours
Radio Interface	Speed:250Kbps (2Mbps), Range: 20 meters
Message	Size: 500KB-1MB, Interval: 1 per minute, TTL: 500 minutes
Buffer size	10-100MB (maximum buffer size that a node is willing to allocate for message distribution)
Nodes	Buses: 8, Cars: 20, Pedestrians: 72, Total: 100 (any node can be source as well as destination)
Node average Speed	Buses: 10-35 km/hour, Cars: 10-50 km/hour, Pedestrians: 0-5 km/h
Mobility Scenario	City environment

Based on several empirical tests, the optimal number of entries per node for time-series data is set as 50. The performance metrics considered for simulations are **(a)** message delivery ratio, **(b)** latency average, and **(c)** overhead.

$$Message\ delivery\ ratio = \frac{1}{M}\sum_{k=1}^{M} R_k\ . \tag{7.3}$$

Message delivery ratio is the percentage of messages delivered successfully. In (7.3), M is total messages created, and $R_k = 1$ if message m_k is delivered, otherwise $R_k = 0$.

$$Latency\ average = \frac{1}{\mathcal{M}}\sum_{k=1}^{\mathcal{M}} Receive\ Time_k - Creation\ Time_k\ . \tag{7.4}$$

In the above equation, parameter $\mathcal{M}$ is the total number of messages received. A message’s latency is the total time spent between message creation and delivery to the destination. Overhead indicates the number of extra transmissions for each delivered message, and is given as:

$$Overhead = \frac{Total\ msgs\ relayed - Total\ msgs\ delivered}{Total\ msgs\ delivered}. \qquad (7.5)$$

To evaluate the performance, the *APS* protocol was compared with four state-of-the-art routing schemes namely: **(a)** *PRoPHET* [7.5], **(b)** *Epidemic* [7.4], **(c)**. *Random* [7.4], and **(d)** *Wave* [7.6]. Simulation results with the Scenario-1 are reflected in Fig. 3(a)–Fig. 3(c), whereas Fig. 3(d)–Fig. 3(f) present the simulation results of Scenario-2. As shown in Fig. 3(a)–Fig. 3(c), the *APS* protocol outperforms the rest of the routing schemes in terms of delivery ratio, latency, and overhead. It is because the *APS* protocol accurately forecasts future contacts by performing online analysis of time-series data of previous contacts. On the contrary, the *PRoPHET* protocol performs future contact estimation on the basis of *number of contacts* instead of duration of contacts. Flooding in the *Epidemic* protocol results in greater message drop due to buffer overflows in resource constrained networks. In contrast, the *APS* protocol performs selective message replication and utilizes the least amount of resources that results in improved delivery ratio. The *Random* protocol forwards single-message copy to any of the randomly selected neighbour, whereas, the *Wave* protocol performs controlled flooding. Although both of the protocols are resource conserving, they show lower performance than *APS*. This is mainly because the aforementioned protocols do not consider past contact patterns to estimate message routes.

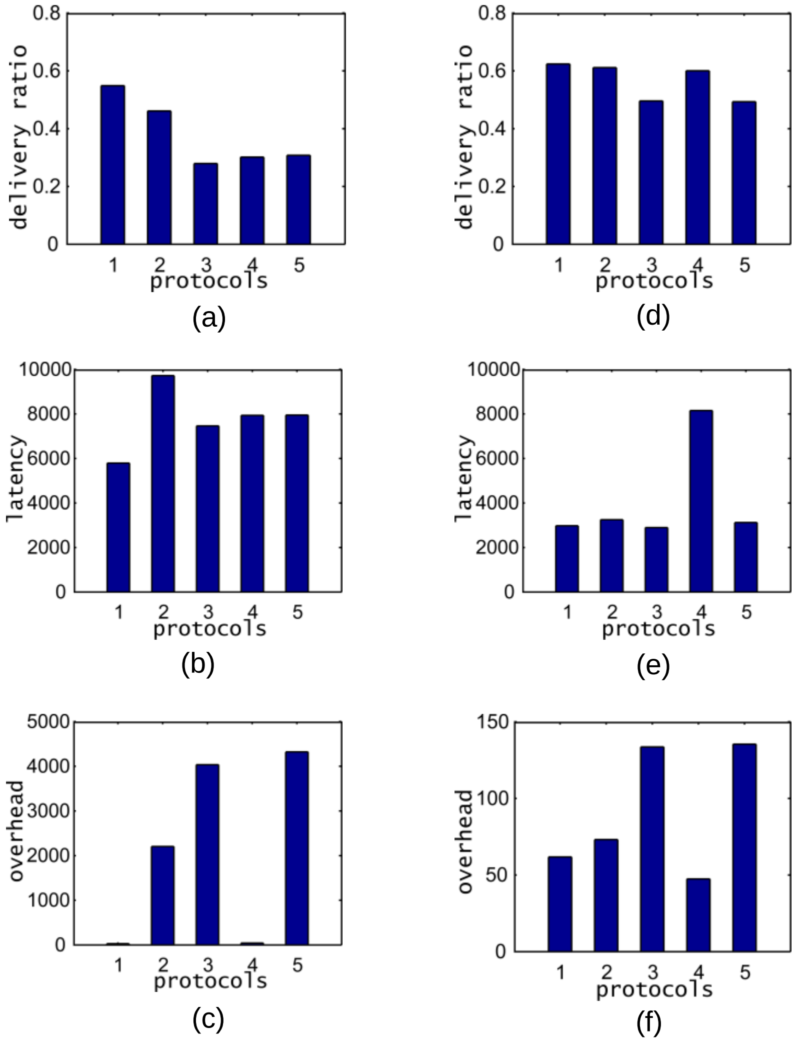


Fig. 7.3. Performance comparison results with scenario-1, (a)–(c) and scenario-2, (d)–(f). The protocols compared are: 1. APS, 2. PRoPHET, 3. Epidemic, 4. Random, and 5. Wave.

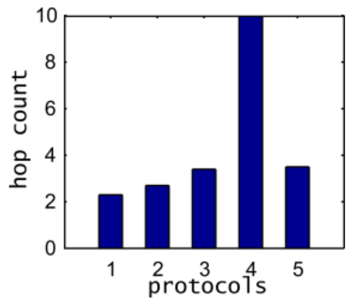


Fig. 7.4. Average hop count for: 1. APS, 2. PRoPHET, 3. Epidemic, 4. Random, and 5. Wave.

As shown in Fig. 3(d)–Fig. 3(f), even by increasing the map size, the performance of the *APS* protocol is better than rest of the schemes, proving that the *APS* protocol is scalable. The latency metric of the *Epidemic* protocol is better than the *APS* protocol (Fig. 3(e)), but at the expense of the *Epidemic* protocol's low delivery ratio. Moreover, the *APS* protocol exhibits lower overhead as compared to the *PRoPHET, Wave*, and *Epidemic* protocols. Fig. 7.4 indicates that despite being multiple-copy replication scheme, the *APS* protocol has lower hop-count average than the rest of the routing schemes.

7.5. References

[7.1] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz, "Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges," IEEE Communications Surveys and Tutorials, vol. 14, no. 2, pp. 607-640, 2012.

[7.2] M. Conti and M. Kumar, "Opportunities in Opportunistic Computing," *IEEE Computer*, vol. 43, no. 1, pp. 42-50, 2010.

[7.3] C. P. Mayer, "Hybrid Routing in Delay Tolerant Networks," *KIT Scientific Publishing*, 2012.

[7.4] A. Balasubramanian, B. Levine, and A. Venkataramani, "Replication Routing in DTNs: A Resource Allocation Approach," *IEEE Trans. Netw.*, vol. 18, no. 2, pp. 596-609, 2010.

[7.5] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "Probabilistic Routing Protocol for Intermittently Connected Networks," http://tools.ietf.org/html/rfc6693, 2012, accessed on Feb 2014.

[7.6] J. Ott, A. Keränen, and E. Hyytiä, "BeachNet: Propagation-based Info-Sharing in Mostly Static Networks," In *Proc. ExtremeCom.*, 2011.

[7.7] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *Technical Report CS-200006, Duke University*.

[7.8] T. Spyropoulos, T. Turletti, and K. Obraczka, “Routing in Delay-Tolerant Networks Comprising Heterogeneous Node Populations,” *IEEE transactions on mobile computing,* vol. 8, no. 8, pp. 1132-1147, 2009.

[7.9] Z. Guo, B. Wang, and J. Cui, “Generic prediction assisted single-copy routing in underwater delay tolerant sensor networks,” *Ad Hoc Networks*, vol. 11, no. 3, pp. 1136–1149, 2013.

[7.10] I. Rhee, M. Shin, S. Hong, K. Lee, S.J. Kim, and S. Chong, “On the Levy-Walk Nature of Human Mobility Networking,” *IEEE/ACM Trans on Networking*, vol. 19, no. 3, pp. 630-643, 2011.

[7.11] C. Song, Z. Qu, N. Blumm, and A. Barabási, “Limits of Predictability in Human Mobility,” *Sci. Mag*, vol. 327, no. 5968, pp.1018-1021.

[7.12] Forecasting: principles and practice, http://otexts.com/, accessed on March 2014.

[7.13] UMass DieselNet trace, http://traces.cs.umass.edu/index.php/Network /Network, accessed on Mar, 2014.

[7.14] The R Project for Statistical Computing, http://www.r-project.org/.

[7.15] CRAWDAD, http://crawdad.cs. dartmouth.edu, accessed on Mar, 2014

[7.16] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE simulator for DTN protocol evaluation,” *2nd Intl. Conf. Sim. Tools. Tech.* 2009.

8. OMNISUGGEST: A UBIQUITOUS CLOUD BASED CONTEXT AWARE RECOMMENDATION SYSTEM FOR MOBILE SOCIAL NETWORKS

This paper is published in *IEEE Transactions on Services Computing*. The authors of this paper are Osman Khalid, Muhammad Usman Shahid Khan, Samee U. Khan, and Albert Zomaya.

8.1. Overview

The advancement in communication infrastructure and easy access of e-commerce and mobile social network applications, such as Amazon, Facebook, Twitter, Foursquare, Instagram, and Path, have shifted the main problem of information retrieval to the filtering of pertinent information [8.1]. The increase in the sheer volume of data with ever-growing networking of devices and Web services has made it quite difficult for users in general to find and access relevant personalized information [8.1].

Recommendation systems were developed in 90s to address the challenges of automatic and personalized selection of data from diverse and overloaded sources of information [8.1]. These systems apply numerous knowledge discovery techniques on users' historical and contextual data to suggest information, products, and services that best match the user's preferences. A good example of recommendation systems for e-commerce applications is Amazon.com, where customers receive personalized recommendations on a variety of products.

8.1.1. Motivation

In the past few years, several social networking applications, such as *Foursquare*, *Gowalla*, and *Google Latitude* were developed for mobile devices. These applications allow users to perform a "check-in" at venues that users visit to share experiences in the form of a feedback or *tip* [8.2], [8.3]. Moreover, these services collect and hold huge volumes of users' geospatial check-in data [8.2]. Based on the data extracted by the mobile social networking applications, several location-based recommendation systems were developed in the recent years [8.1], [8.2], [8.3], [8.4], which recommend venues to users closely related to their preferences. A major research challenge for such systems is to generate real-time venue recommendations for a given individual from a massively diverse dataset of users' historical check-ins [8.1], [8.3], [8.4]. To generate an optimal recommendation for an individual, the system must simultaneously consider the following factors: **(a)** personal preferences, **(b)** past check-ins, **(c)** current context, such as time and location, and **(d)** collaborative social opinions (other individuals' preferences).

The objective of this chapter is to efficiently employ the above mentioned factors to achieve real-time, optimal recommendations for venues. However, there are several barriers that negatively affect the performance of real-time recommendation process primarily driven by the complexity and cost of processing the large-scale data sets [8.1], [8.2]. To scale efficiently, the recommendation system requires large-scale computational and storage resources. This chapter describes an approach that leverages cloud infrastructure and service-based interfaces to process, mine, compare, and manage large-scale datasets for real-time recommendations in a scalable architecture.

8.1.2. Research Problem

Several works [8.1], [8.2], [8.3], [8.5], [8.6] have applied collaborative filtering (CF) to the venue recommendation problem. These CF-based venue recommendation systems work by matching a given user's venue check-in record with the other users stored in a user-venue check-in matrix. The objective is to find a subset of *similar users* who share similar tastes and patterns on the basis of visited venues compared to a given user. These similar users share their judgments and opinions on venues, and in return, the system provides useful personalized recommendations for a given user. However, the following unsolved problems of the previous work affect the performance of current venue recommender systems:

- *Data sparseness.* A user may have visited only a limited number of venues, and as a result there would be a sparse user-venue check-in matrix. The data sparseness causes poor calculations of the nearest neighbor set of users based on similarity with current user, which results in the loss of accuracy of recommendations. Moreover, sparseness of matrix results into the suboptimal performance of many existing venue recommendation systems [8.3] that directly apply collaborative filtering based models on user-venue matrix. Apart from venue recommendation systems, the data sparseness also negatively affects item recommendation systems, such as Amazon.com, where active users may have purchased below 1% of the items [8.1], [8.2].
- *Cold start.* The cold start problem in many existing CF recommendation systems [8.3], [8.7] usually occurs when recommendations are to be generated for a user that is new to the system. This is because the system does not have sufficient record available for new user to perform similarity measures. Insufficient records results in the zero values of similarity computations, which degrades recommendation quality.
- *Scalability.* The memory-based CF recommender systems use user rating data to apply simplistic approaches of computing similarity between users or items (such as neighborhood based CF [8.7], [8.8], [8.9]). However, such systems also suffer from scalability issues, as they need to parse thousands of users at real-time in user-venue matrix that is neither efficient nor scalable. To address the scalability issues, a few proposals applied model based CF. The model-based approaches apply data mining and machine learning algorithms to find patterns based on the training data to reduce the size of the user-item rating matrix [8.1], [8.2]. However, there is an inherent tradeoff between

reduced dataset size and recommendation quality. If a dataset is reduced for fast online processing, then it may result in the loss of recommendation quality.

The immediate repercussion of the above listed issues is the suboptimal performance in CF-based recommendation models. Therefore, it may not be feasible to solely use memory-based CF model for venue recommendations.

8.1.3. Contributions

We propose a novel hybrid cloud based venue recommendation framework, *OmniSuggest*, which combines memory-based and model-based approaches of CF on a cloud framework to generate optimal recommendations. To address the problems of data sparseness and cold start, our framework utilizes a model-based Hyperlink-Induced Topic Search (HITS) [8.10] approach to select *popular venues* for each category (e.g., Food) under multiple levels of hierarchies (e.g., Asian Food → Chinese Food). Such a methodology enables our proposed cloud based *OmniSuggest* framework to generate recommendation for a new user through the collaborative opinion of *experienced users (hubs)*, by computing (memory-based) similarities in preferences of new user and the experienced hubs.

Apart from recommendations for an individual user, we propose a method to generate venue recommendations for a group of users or *friends* sharing a common interest. As an example, a group of friends may require the recommendation for “Chinese Food”, and they want to attend dinner together. Moreover, unlike the existing systems [8.4], [8.5], [8.6], [8.7], [8.11], [8.12] when generating recommendation for a whole group, our system also considers the real-time effect of various parameters, such as distance of each group member from a set of top venues, the road traffic conditions, and other obstacles that may be encountered in reaching a venue. To reduce the computational cost during real-time processing, and ensure its 24×7 omnipresence, the cloud based *OmniSuggest* framework follows Software as a Service (SaaS) approach through a modular service based architecture. One of the major advantages of this approach is that the *OmniSuggest* framework can scale on demand as additional virtual machines are created and deployed. In summary, the contributions of our work are:

- A recommendation framework is presented that combines social computing, and recommendation modules, on cloud infrastructure, to ensure scalability in terms of processing, storage, and parallelization. We combine the model-based and memory-based CF algorithms into a hybrid approach that significantly improves the recommendation accuracy compared to previous venue recommendation algorithms.
- To resolve the issues associated with data sparseness and cold start, the proposed framework models the users' data by utilizing HITS method to extract experienced users and popular venues for multiple categories. A variant of Ant colony algorithm is applied to generate a set of venues for a user.
- The cloud based *OmniSuggest* framework performs group recommendations by using a combination of collaborative filtering and group satisfaction principle. The group satisfaction mechanism is implemented as to depict a Service Level Agreement (SLA) between the *OmniSuggest* framework and the end users. The SLA ensures the provision of on time, high quality recommendations, proportionate to the real-time changes (such as, traffic conditions) that occur when group members move towards recommended venue.
- We have carried out experiments on our internal Ubuntu cloud setup running on 96 core Supermicro SuperServer SYS-7047GR-TRF systems. The experiments are conducted on real-world dataset from *Foursquare*.

The remainder of this chapter is organized as follows: The system architecture is described in Section 8.2. In Section 8.3, we present the model for individual and group recommendations. Section 8.4 presents the experimentation results. Related work is discussed in Section 8.5, and Section 8.6 concludes the chapter with a summary and a description of the future work.

8.2. System Architecture

Most of the existing recommender systems are based on centralized architectures [8.1], [8.2], [8.3], [8.8], [8.9], [8.12]. Such systems are not scalable enough to handle large volumes of geographically distributed data. The increasing number of subscribers in mobile social networks puts forth new challenges for centralized systems. Such systems must simultaneously consider a user's preference, social context, and past actions when generating online recommendations. Therefore, to address the scalability issues, we utilize a decentralized cloud based approach.

8.2.1. Major Components

The following are the major components of the proposed cloud based framework:

User profiles. The *OmniSuggest* framework maintains users' profiles that contain information about the venues visited by users. Venues are categorized into various types based on the dataset analysis of location-based services, such as *Foursquare* and *Gowalla*. For example, in Fig. 8.1, the parent category "Food" has two sub-categories at level-1: $\boldsymbol{A}$ (e.g., Asian Food) and $\boldsymbol{B}$ (e.g., American Food). Category $\boldsymbol{A}$ is further having three sub-categories at level-2: $\boldsymbol{A_1}$ (e.g., Chinese Food), $\boldsymbol{A_2}$ (e.g., Thai Food), and $\boldsymbol{A_3}$ (e.g., Indian Food). The categories at Level-1 and Level-2 are associated with venues. The arrows from users indicate the number of check-ins performed by users at various venues.

The cloud based *OmniSuggest* framework (Fig. 8.2) maintains the categories up to two levels to ensure the finer granularity of information. Moreover, each check-in record has the following fields: **(a)** user identification, **(b)** venue name and identification, **(c)** venue location (GPS location, city, and country), **(d)** time at which user performed check-in at a venue, and **(e)** parent and sub-categories a venue is associated with. User profiles are geographically distributed on the basis of cities. As depicted in Fig. 8.1, for each geographic region, the framework maintains a record of the sets of venues checked-in by users under each category in the hierarchy.

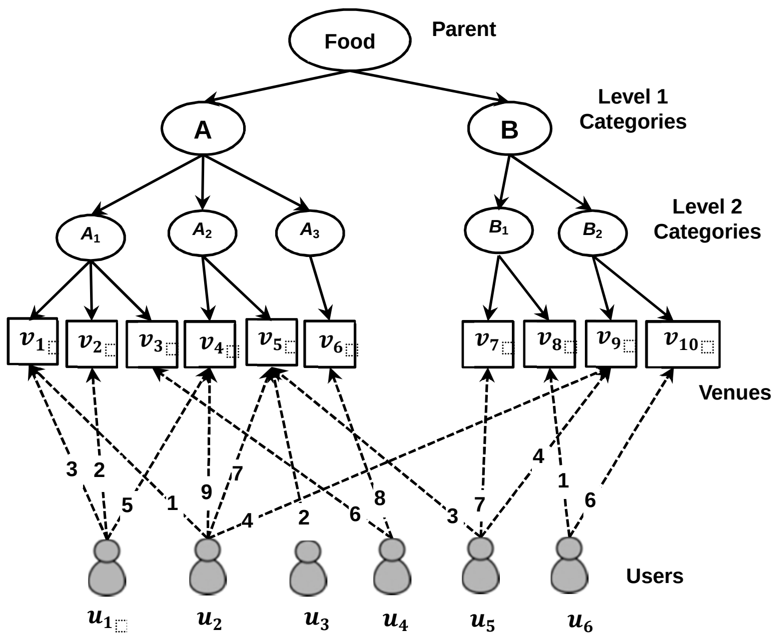


Fig. 8.1. Venues are linked with various categories at multiple levels. The lower half indicates users who have performed check-ins at venues. A venue may be linked with multiple categories.

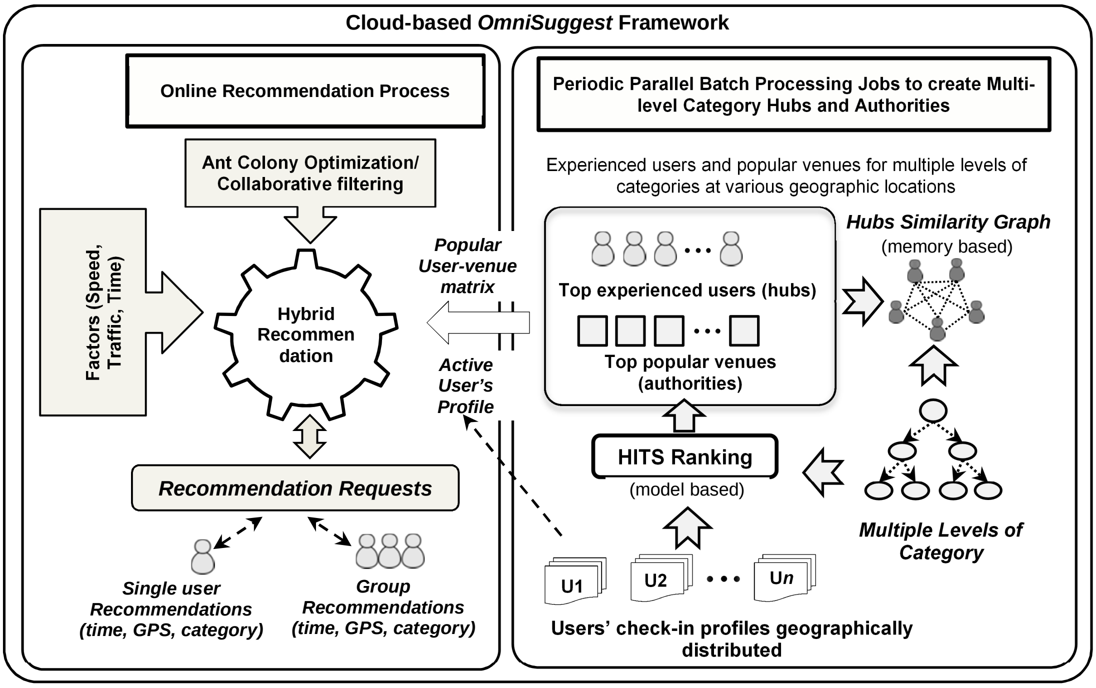


Fig. 8.2. A top level architecture of the Cloud-based OmniSuggest framework.

- *Top-K Users and Venues.* The proposed framework employs HITS method [8.10] on user profiles to generate experienced users and popular venues for multi-level categories under each parent category. The HITS approach gives higher popularity ranking to a user if (s)he visits a set of venues that are most frequently visited. Similarly, a venue is ranked higher, if it is most frequently visited by experienced users. As an example, the framework maintains user-venue popularity sets for categories: "Chinese Food", "Asian Food", or just "Food". The framework also computes similarity graphs among experienced users in various categories' hierarchies. The similarity graphs computed during offline phase are later utilized during online recommendation phase. The HITS-ranking for users' and venues' is stored in the framework's geographically distributed databases. Such a methodology further helps distribution and parallel execution of processing tasks on cloud framework as each place has local sets of users and venues.

- *Recommendation Module.* The recommendation module performs parallel execution of venue recommendation requests generated by an *active user* or a *group* of friends. The recommendation request query consists of current time, GPS location, and category for which the active user requires top-*N* venue recommendations. For a given category type, the recommendation framework pulls out similarity graph of top-*E* experienced users, where *E* is the number of experienced users. A modified version of Ant colony algorithm and collaborative filtering is then applied to generate an optimal solution in the form of venues that best match an active user's preference. While generating the ranking of venues for the group, the recommendation framework also takes into account the effect of real-time factors (speed, distance, and road-traffic conditions). Therefore, to ensure the SLA is properly abided by the *OmniSuggest* framework, the venue at top of the ranked list will be the one that satisfies all of the group members.

The *OmniSuggest* framework runs HITS method and computation of experienced users' similarity graph as periodic batch processing jobs on users' profiles. These jobs are meant to refine data through preprocessing and prune the insignificant entries. Moreover, such jobs can be scheduled to run during off-peak load hours in various geographical regions to reduce unnecessary computational burden on the cloud nodes.

8.2.2. Cloud Services Mapping

As reflected in Fig. 8.3, OmniSuggest framework follows a SaaS approach through a modular service based architecture. The SaaS forms the top layer of the cloud stack, offering real-time personalized recommendations to a user or groups of users, while abstracting underlying implementation details [8.13]–[8.15]. Users access the service using thin clients, such as mobile devices, and are typically unaware of the physical locale of the hosted service. The framework ensures that the SLA is maintained by generating real-time context-aware recommendations that increase the global satisfaction of group members on the recommended venues. Moreover, the configuration allows the framework to scale on demand as additional virtual machines are created and deployed to handle the sporadic requests from the users.

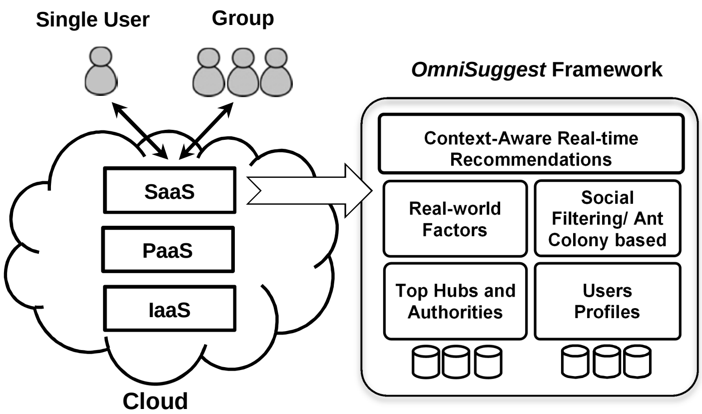


Fig. 8.3. OmniSuggest framework's cloud services mapping.

8.3. Proposed Recommendation Framework

In this section, we discuss in detail the proposed cloud based venue recommendation framework, *OmniSuggest*. In terms of functionality, *OmniSuggest* framework has two main modules: **(a)** an offline processing module and **(b)** an online recommendation module. The offline processing module runs periodic jobs to pre-process the check-in data. Data pre-processing involves two phases: **(a)** popularity ranking of users and venues, and **(b)** similarity graph creation among popular users. The online recommendation module is responsible for generating the recommendations for an individual user or a group of friends. The detailed functionality of the above mentioned modules is discussed in the following subsections. We use the following notations in rest of the chapter: G_c (hubs similarity graph for category c), N (number of venues recommended), V (set of all venues), U (set of all users), and v_{ix} (number of check-ins of a user i to a venue x), and v_i (total number of check-ins of user i).

8.3.1. Offline Preprocessing

8.3.1.1. User Venue Popularity Ranking

This subsection presents the methodology of assigning popularity ranking to users and venues for various category hierarchies in a geographic location. The HITS [8.10] mechanism is utilized to perform the ranking for producing a set of experienced users and popular venues. In Fig. 8.1, suppose we want to calculate the hub and authority scores for category *A* under the parent category ***Food.*** First, we need to create a user-venue matrix for category ***A.*** Let the matrix be represented as M_A, having *U* rows and *V* columns. Let $[h_A]$ and $[a_A]$ represent the hub and authority score matrices, respectively for a category ***A.*** The following formulas compute the hub and authority scores [8.10].

$$a_A = M_A^T \times h_A. \tag{8.1}$$

$$h_A = M_A \times a_A. \tag{8.2}$$

If we use $a_A^{<n>}$ and $h_A^{<n>}$ to represent the hub and authority scores at *nth* iteration, then following are the equations for generating the hub and authority scores.

$$a_A^{<n>} = (M_A^T \times M_A) \times a_A^{<n-1>}. \tag{8.3}$$

$$h_A^{<n>} = (M_A \times M_A^T) \times h_A^{<n-1>}. \tag{8.4}$$

The insight into using the HITS method is to generate a subset of users, who have higher experience of visiting popular venues, and a subset of venues, that are being frequently visited by the experienced users. We call such subsets as popular authorities and experienced hubs. The hub and authority scores are computed as batch processing jobs separately for each of the individual category. Therefore, the scores, and the iterations (*n*), vary from category to category. The following are the number of iterations required to converge the scores for the sample categories presented here, as an example: American Food: n=56, Chinese Food: n=51, and Thai Food:

n=945. We do not store users/venues with very low HITS scores in the database of experienced users and popular venues. This helps in avoiding unnecessary computations during the online recommendation.

8.3.1.2. Hubs Similarity Graph Creation

This phase creates similarity graphs among experienced users (hubs) under the various predefined categories. The idea is to generate a network of like-minded people who share the similar preferences for various venues they visit in a geographical region. The graphs constructed in current phase will be made available for online recommendation process that utilizes a variant of Ant colony algorithm to find an optimal path on the graph. Such a path carries a collective opinion about venues by experienced users who are also most similar to an active user.

The similarity computation between two users in the hub similarity graph is performed by applying the Pearson Correlation Coefficient (PCC)[8.1]. The value of PCC ranges between -1 and +1. Positive values indicate that the similarity exists between two users, with highest similarity at 1, whereas negative PCC values means the choices of the two users does not match. PCC is computed by using the following formula.

$$sim(i,j) = \frac{\sum_{x \epsilon S_{ij}} (v_{ix} - \overline{v_i})(v_{jx} - \overline{v_j})}{\sqrt{\sum_{x \epsilon S_{ij}} (v_{ix} - \overline{v_i})^2 \sum_{x \epsilon S_{ij}} (v_{jx} - \overline{v_j})^2}}, \tag{8.5}$$

where

$$S_{ij} = \{x \epsilon V | v_{ix} \neq 0 \wedge v_{jx} \neq 0\}.$$

In (8.5), the similarity between two users i and j is computed only for venues that are visited by both of the users. Moreover, an edge is created between the two users in the graph, if their PCC value is positive. Considering only the positive PCC values may result into a very sparse similarity graph among the experienced users, when the user-venue check-in matrix is already very sparse. To address the sparseness issue we augment similarity computation with the conditional probability. The conditional probability computes the likeliness that a venue will be visited by both users i and j. Moreover, it depicts the amount of interest (or confidence) showed by both users in venues commonly visited by them. The following equation is defined to calculate the weight of an edge between two users.

$$\omega_{ij} = \begin{cases} sim(i,j) & \text{if } sim(i,j) > 0 \\ & \text{otherwise} \\ \dfrac{P[v_i \cap v_j]}{P[v_j]} \times \dfrac{1}{1 + \sum_{x \in V_j} |v_{ix} - v_{jx}|} & P[v_j] \neq 0, \end{cases} \tag{8.6}$$

where V_j is the set of venues checked-in by user j. In (8.6), positive values of similarity are given preference over conditional probability. Moreover, denominator value of the conditional probability postulates the edge between two users to be a directed edge. The additional sum factor in denominator is to decrease the value of conditional probability to keep it lower than similarity. The region-wise similarity graphs among the experienced hubs for various categories are stored in the database for the online recommendation process.

8.3.2. Online Recommendation for Single User

In this subsection, we present the online recommendation framework that applies a variant of the Ant colony approach on a graph of experienced users (hubs) to generate a set of the most popular venues not previously visited by an active user. Most of the popular collaborative filtering techniques, such as the ones we used for evaluations, are greedy based [8.1], [8.2], [8.4], [8.5], [8.9], [8.16]. Intuitively, the greedy based heuristics are very efficient, but they may not be very effective. The main limitations of such approaches are that they provide recommendations solely based on the opinion of the users who are most similar to the current active user. However, a user who is most similar to the active user may not have necessarily visited most of the new venues that we want to recommend to the active user. To address such limitation, we used a meta-heuristic that had the capability of backtracking. For that purpose, we chose the Ant colony approach, where we applied a pheromone update strategy that iteratively updates pheromone values of the graph edges. As the iterations proceed, the pheromone concentration increases on the edges that lead towards the nodes that are not only the most similar nodes to the active user, but also provide maximum contribution of the venues that needs to be recommended to the active user. Algorithm 8.1 illustrates the procedure of the online recommendations.

Algorithm 8.1. Ant Colony based Venue Selection

Input: Active user: s, category: $\mathcal{C}$, region: R

Output: A set V' of top-N venues visited by experienced hubs similar to active user.

Definitions: *t = Current Iteration, t_{max} = maximum iterations, N_j=neighbor set of node j, $\tau(i,j)$=sim(i,j) if δ_{ij}=1 $\wedge$ $i = s$, otherwise, $\tau(i,j) = \omega_{ij}$ (from (8.6)), where δ_{ij} = edge count between i and j, $\eta(i,j) = 1/\delta_{ij}$ and, $\mathcal{Z}_j$= number of required venues found at a node j.*

1: $t \leftarrow 0$; $a \leftarrow s$; $\delta \leftarrow 1$; $level_k \leftarrow \emptyset$; $edges_k \leftarrow \emptyset$
2: $G_c \leftarrow getHubSimGraph(\mathcal{C}, R)$
3: $N_a \leftarrow \{x: G_c | sim(a,x) > 0\}$
4: $k \leftarrow |N_a|$ number of ants
5: $t \leftarrow t + 1$
6: $tabu_k \leftarrow a$
7: Sort N_a in terms of $[\tau(a,j) \times \eta(s,j)]$, $j \in N_a$ (descending)
8: **for** each $e \in N_a$ **do**
9: $\quad S \leftarrow \{v: V_e | v \notin V_a\}$
10: $\quad M \leftarrow M.append(e, S)$
11: $\quad tabu_k \leftarrow tabu_k \cup \{e\}$
12: $\quad edges_k \leftarrow edges_k \cup \{\tau(a,e)\}$
13: **end for**
14: **if** $venueCount(M) \geq N$ **then**
15: $\quad$ go to Line 25
16: **else**
17: $\quad \forall j \in N_a$, select $a \leftarrow j$, such that we have $arg\,max\left[\tau(a,j) \times \eta(s,j) \times \frac{\mathcal{Z}_j}{N}\right] \wedge N_j \neq \emptyset \wedge \forall g \in N_j |\ g \notin tabu_k$
18: $\quad$ **if** $No\ any\ such\ node\ found\ in\ Step$ 17 **then**
19: $\quad\quad$ go to Line 25
20: $\quad$ **else**
21: $\quad\quad \delta \leftarrow \delta + 1$; $level_k.append(a)$
22: $\quad\quad$ go to Line 7
23: $\quad$ **end if**
24: **end if**
25: *evaporate_deposit_Pheromone()*
26: **if** $t \leq t_{max}$ **then**
27: $\quad$ Reset $tabu_k, level_k, M$, and set $\delta \leftarrow 1$
28: $\quad$ go to Line 4
29: **else**
30: $\quad V' = aggregate(M)$
31: **end if**
32: **return** V'

1. *Initializations (Line 1–Line 3):*

    - The algorithm takes as input the following parameters: **(a)** identification of the active user, **(b)** the category for which the active user wants recommendations, and **(c)** geographical region where the user is currently located. For example, an active user "*s*" is interested in "Chinese Food" and located in the New York City. In Line 1, various data structures used by the algorithm are initialized. The graph of experienced hubs for the specified category and geographic region is retrieved from the database in the Line 2. In Line 3, the links of the active user are created with the subset of graph nodes (N_a) based on the similarity formula (8.5).

2. *Iterative solution construction (Line 4–Line 29):*

    - The algorithm increments the iteration counter (t) after creating the ants, and inserts the entry of the active user in the tabu-list of ant k (Line 4–Line 6).
    - The neighbor nodes (N_a) are traversed in the descending order based on the existing pheromone quantity on the links, multiplied by the edge count between the active user and neighboring node (Line 7–Line 8).
    - On the traversal, only those venues are collected from the neighboring nodes that were not previously visited by the active user (Line 9).
    - The collected venues are appended in a matrix (M) (Line 10). The visited neighbor, as well as, the pheromone on the edge is stored in the respective lists (Line 11–Line 12).
    - The Line 14 checks whether or not the required number of venues have been collected. Two cases may arise: **(a)** venue count has reached N and **(b)** required venue count has not been achieved.

(a) If the venue count is achieved, then the control parses the Line 25, where the pheromone is updated on the graph edges (discussed subsequently). After that, it is checked whether or not, the maximum number of iterations has been reached (Line 26). In the case when the current iteration count (*t*) is less than the maximum allowed iterations (t_{max}), then the data structures will be reset (Line 27) and the control jumps to Line 4. Otherwise, if the test condition at Line 26 is false, then this means that the maximum number of allowed iterations is completed, and the venues are ranked using the aggregation function (discussed later subsequently).

(b) If the required venue count is not achieved, then the control jumps to Line 17, where a node is selected amongst the neighbor set (N_a). The criterion for the node selection is that the maximum pheromone must be deposited on the link towards selected node, and the selected node has the maximum number of venues available for the active user. If no such node is found, then this means that the ant has reached the terminal node of the graph. Subsequently, the control will jump to Line 25. Otherwise, the selected node will be set as a new temporary active user (*a*) and appended in the list (Line 21). Moreover, the edge count will also be incremented by one in Line 21. From Line 22, the control will jump back to Line 7, and the procedure will be repeated iteratively until the maximum number of iteration limit is reached.

3. *Pheromone Update (Line 25):*

- The pheromone is updated in two steps: **(a)** evaporation and **(b)** deposition. Evaporation is performed equally on all the edges of the graph. However, only the edges leading to the nodes that provide the required venues are deposited with the pheromone. The pheromone deposit depends on: **(a)** the existing quantity of pheromone between the node *j* and active user *s*, **(b)** number of venues returned by the node *j*, **(c)** the hop distance between the node *j* and the user *s*, and **(d)** average check-ins of node *j* at venues retrieved in current iteration.

- For the iteration *t*, the pheromone is evaporated on each edge at a rate, given by $(1-\rho) \times \tau_{ij}(t-1)$, where ρ is a constant that represents the evaporation rate. The amount of pheromone deposited on an edge is given as:

$$\Delta\mathcal{D}_{sj}(t) = \frac{\prod_{r=1}^{\delta_{sj}} \tau_{r,r+1}(t-1)}{\delta_{sj}} \times \frac{\mathcal{Z}_j}{N} \times \frac{\sum_{x \in S'} v_{jx}}{\sum_u \sum_{x \in S'} v_{ux}},$$

where $\prod_{r=1}^{\delta_{sj}} \tau_{r,r+1}(t-1)$ represents the product of pheromone deposited on the edges between node *s* and node *j* that are δ_{sj} units apart, $S' = \{V_j \backslash V_s\}$, and $u \in tabu_k$. The parameter $\frac{\mathcal{Z}_j}{N}$ indicates the ratio of the number of venues contributed by a node *j*, to the total number of the required venues (*N*). The right most term in multiplication indicates the average number of check-ins performed by the user *j* (per iteration) at venues not visited by an active user *s*. For every iteration, the values of the pheromone at time $(t-1)$, and the nodes that were selected as temporarily active user (*a*) at each level of the graph, are stored in the data structure $edges_k$ and $level_k$. The aforementioned data structures provide the necessary values required to update the pheromone in the current iteration. Therefore, the aggregate quantity of pheromone updated at the end of iteration *t* is given by:

$$\tau_{ij}(t) = (1-\rho) \times \tau_{ij}(t-1) + \Delta\mathcal{D}_{sj}(t) \tag{8.7}$$

4. Aggregate venues provided by the best nodes (Line 30):

- On completion of t_{max} iterations, the venues are ranked and sorted in the descending order to generate top-N venues to be recommended to the active user. The following equation is used to rank the venues.

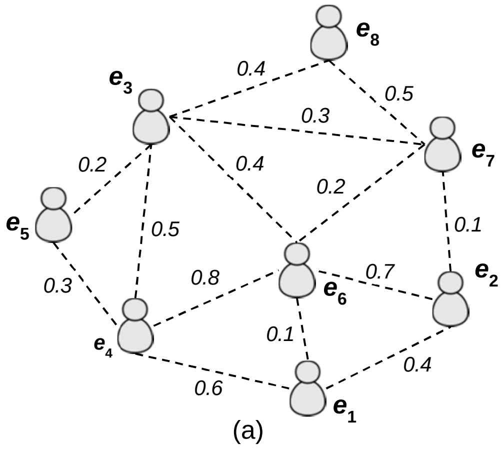
$$Rank_x = \frac{\sum_u \tau_{su}(t) \times v_{ux}}{\sum_u \tau_{su}(t)}. \tag{8.8}$$

In (8.8), x is the venue to be ranked, the parameter s is the active user node, and v_{ux} is the number of check-ins performed by user u at venue x. The parameter $\tau_{su}(t)$ represents the quantity of pheromone between node s and user u after t_{max} iterations.

It is noteworthy to mention that by using (8.8), we can observe that the quantity of pheromone accumulated on the edges after multiple iterations has a significant effect on the ranking of the venues.

8.3.2.1. An Illustrative Example

Suppose that by using Algorithm 8.1, we have to recommend ten venues to an active user s under a specific category. The venues to be recommended are the ones not previously visited by the active user. As a first step, the graph of experienced users under a given category will be retrieved from the database as depicted in Fig. 8.4(a). The similarity of the active user will be computed with all of the nodes in the graph. New links will be created between the active user and only those graph nodes for which the similarity is greater than zero.



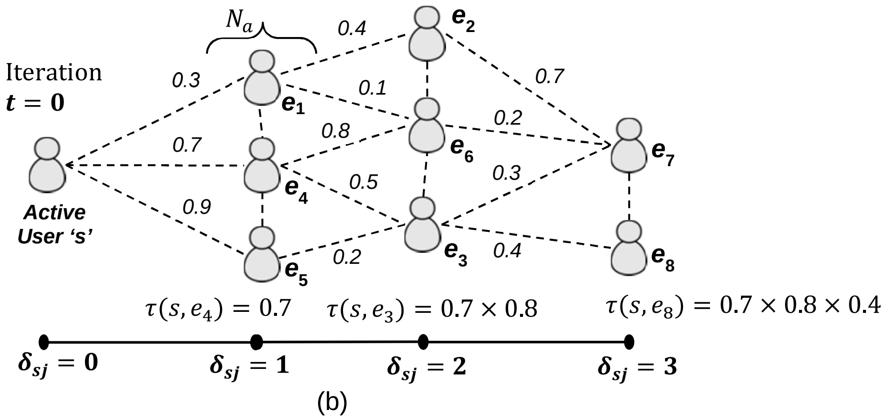


Fig. 8.4. (a) Hubs similarity graph retrieved from database, and (b) connectivity of active user with hub similarity graph.

By setting the active user s as the root node, the Breadth First Search (BFS) procedure will be applied to place the immediate neighbors of the active user (for which similarity is greater than zero) at a distance of one ($\delta_{sj} = 1$), as depicted in Fig. 8.4(b). At a distance of two ($\delta_{sj} = 2$) the neighbors of the friends of the active user will be placed, and the process continues until the whole of the graph is traversed. As indicated in Fig. 8.4(b), each edge has a weight, and the edges connecting the nodes at the same level of the distance are intentionally labeled blank due to the fact that they are not traversed during the execution of Algorithm 8.1. Suppose that the ten venues that we want to recommend to the active user are shown as column labels in Table 8.1. The entries in each of the columns reflect the number of check-ins performed by the hub users at the visited venues. The last column indicates the total number of venues visited by each of the hub user out of the required ten venues. On the execution of Line 8–Line 13 in Algorithm 8.1, the ant collects the venues from the neighbors of the active user. The number of venues collected from each of the neighbor are $e_5(1)$, $e_4(3)$, and $e_1(2)$. (Despite the fact that e_5 is the most similar to the active user, the node e_5 was able to produce only a single venue.) The execution jumps to Line 17, where the best of the visited neighbor nodes is selected based on the pheromone value, and the number of venues contributed by the neighbor node. Consequently, we get the following values for each neighbor: e_5[0.9×1×(1/10) = 0.09], e_4[0.7×1×(3/10) = 0.21], and e_1[0.3×1×(2/10) = 0.06]. Here, e_4 will be selected as the new root (a) node because of having highest the value. Line 8–Line 13 will be executed again and the venues collected from the neighbors are: $e_6(2)$ and $e_3(3)$. On the execution of Line 17, the following values are obtained for each of the neighbor: e_6[0.8×(1/2)×(2/10) = 0.08] and e_3[0.5×(1/2)×(3/10) = 0.075]. Therefore, e_6 will be selected as a new root node (a). After Line 8–Line 13 are executed, the venues collected from the neighbors are $e_7(3)$. The node e_7 does not have any further neighbors (Line 17).

Table 8.1. Number of times required venues are visited by each hub user and total check-ins at the venues.

	v_1	v_3	v_4	v_7	v_{11}	v_{12}	v_{22}	v_{25}	v_{44}	v_{45}	z_j
e_1	4	-	-	-	-	-	-	-	7	-	**2**
e_2	-	-	-	-	-	-	-	-	-	-	**0**
e_3	-	29	12	47	-	-	-	-	-	-	**3**
e_4	-	-	14	-	-	11	18	-	-	-	**3**
e_5	-	6	-	-	-	-	-	-	-	-	**1**
e_6	-	-	-	-	14	-	19	-	-	-	**2**
e_7	55	2	-	7	-	-	-	-	-	-	**3**
e_8	-	-	-	-	-	-	-	40	-	30	**2**

Table 8.2. Pheromone update on edges.

$(1-\rho)\times\tau_{ij}(t-1)$ $\rho = 0.01$	Path followed from node s to node j	$\prod_{r=1}^{\delta_{sj}}\tau_{r,r+1}(t-1)$	$\mathcal{Z}_j/N$	$\frac{\sum_{x\in S'} v_{jx}}{\sum_u \sum_{x\in S'} v_{ux}}$	$\Delta\mathcal{D}_{sj}(t)$	$\tau_{ij}(t)$
$0.99\times\tau(s,e_1)$ $= 0.99\times 0.3 = 0.297$	$s-e_1$	0.3	2/10	11/245	0.3×(2/10)×(11/245)=0.0026	0.2996
$0.99\times\tau(s,e_4)$ $= 0.99\times 0.7 = 0.693$	$s-e_4$	0.7	3/10	43/245	0.7×(3/10)×(43/245)=0.0368	0.7298
$0.99\times\tau(s,e_5)$ $= 0.99\times 0.9 = 0.891$	$s-e_5$	0.9	1/10	6/245	0.9×(1/10)×(6/245)=0.0022	0.8932
$0.99\times\tau(e_1,e_2)$ $= 0.99\times 0.4 = 0.396$	-	-	-	-	-	0.396
$0.99\times\tau(e_1,e_6)$ $= 0.99\times 0.1 = 0.099$	-	-	-	-	-	0.099
$0.99\times\tau(e_4,e_6)$ $= 0.99\times 0.8 = 0.792$	$s-e_4-e_6$	0.7×0.8=0.56	2/10	33/245	(0.56/2)×(2/10)×(33/245)=0.0075	0.7995
$0.99\times\tau(e_4,e_3)$ $= 0.99\times 0.5 = 0.495$	$s-e_4-e_3$	0.7×0.5=0.35	3/10	88/245	(0.35/2)×(3/10)×(88/245)=0.0188	0.5139
$0.99\times\tau(e_5,e_3)$ $= 0.99\times 0.2 = 0.198$	-	-	-	-	-	0.198
$0.99\times\tau(e_2,e_7)$ $= 0.99\times 0.7 = 0.693$	-	-	-	-	-	0.693
$0.99\times\tau(e_6,e_7)$ $= 0.99\times 0.2 = 0.198$	$s-e_4-e_6-e_7$	0.7×0.8×0.2=0.112	3/10	64/245	(0.112/3)×(3/10)×(64/245)=0.0029	0.2009
$0.9\times\tau(e_3,e_7)$ $= 0.99\times 0.3 = 0.297$	-	-	-	-	-	0.297
$0.9\times\tau(e_3,e_8)$ $= 0.99\times 0.4 = 0.396$	-	-	-	-	-	0.396

Therefore, the condition of Line 18 will become true and the control will jump to Line 25, where the pheromone values on the edges will be updated. The process will be restarted from the actual root node (s), and will be repeated for t_{max} iterations. Table 8.2 shows the update of the pheromone values after the first iteration. It can be observed that after the first iteration is completed, the pheromone on the edge e_4–e_6 has been decreased; whereas, the pheromone value is increased on the edge e_4–e_3. Therefore, after several iterations ($<t_{max}$), the current path s–e_4–e_6–e_7 will be replaced by new paths s–e_4–e_3–e_7 and s–e_4–e_3–e_8, which will also increase the number of venues collected for the active user.

8.3.3. Group Recommendation

The existing work on venue recommendation systems, generally, focuses on recommending venues to individual users based on personal preferences [8.1], [8.2]. However, the process becomes quite challenging when the system must provide recommendations to a group of friends [8.11], [8.17]. To achieve an optimal level of satisfaction for the whole group, the recommendation system must be able to conglomerate the consent of all of the group members [8.23]. However, it is not a simple task to produce recommendations that satisfy every group member, as an individual's context (e.g., speed, distance, and road traffic conditions) may vary with time [8.11].

To address the above mentioned issues, we propose a more efficient and effective approach for group recommendations. The proposed cloud based *OmniSuggest* framework also takes into consideration the effect of various real-world time-varying factors, such as speed, distance, and traffic conditions, on the group recommendations. In the following subsection, we present a motivational example that highlights the problems faced by a "traditional" venue recommendation system which generates recommendations without considering the aforementioned real-world factors (e.g., speed, distance, and road traffic conditions). Later, we present a technique to circumvent the anomalies associated with the traditional venue recommendation systems.

8.3.3.1. A Motivational Scenario

Suppose, a group of five friends are at different parts of a city and they decide to get together for dinner. They plan to meet at a Chinese restaurant. One member of the group, known as *group leader*, initiates a group query that consists of: **(a)** deadline by which they must arrive at the venue, and **(b)** identifications of group members. The recommendation system recommends a venue for the category "Chinese Food" based on the popularity ranking calculated for the venue (Section 8.3.1.1) within the geographical region [8.11], [8.23]. Out of the five, two group members are located far away from the recommended venue, and are unwilling to undertake a long journey. One member is stuck in a traffic jam on the road leading to the venue and is unable to reach before the deadline. Only the remaining two members will be able to reach the venue before the deadline. However, they also drop their plan on finding out that the other members are not coming. Despite that recommended venue is most popular the system still could not satisfy the whole group as each member had time-varying context that subsequently developed into constraints.

The above scenario depicts our motivation behind considering real-world parameters (e.g., speed, distance, and road traffic conditions) in the venue recommendation process. When generating recommendations, our proposed framework not only considers the highly ranked venue (using HITS method) for a specific category, but also takes into account the current context of each of the group member. In this fashion, a venue is reported on top of a list by considering the popularity and the mutual consent of the group members. In the following text, we present our group recommendation approach.

Algorithm 8.2. Group Recommendation

Input: Group query G_q, threshold time $\mathcal{T}$, category $\mathcal{C}$, region R

Output: Recommended venue for the group.

$Q \leftarrow$ Retrieve group members from G_q

$V' \leftarrow getTopVenues(\mathcal{C}, R)$

for each member $m \in Q$ **do**

  for each venue $v \in V'$ **do**

    $T_{mv} = \mathcal{T} - \frac{|Loc_m - Loc_v|}{1 + Speed_m \times Road\ Cond_{mv}}$

    $P[m][v] = \begin{cases} T_{mv} \times v_{mv} \times A_v, & \text{if } T_{mv} > 0 \\ 0, & \text{otherwise} \end{cases}$

  end for

end for

$Rank[v] \leftarrow aggregate(P)$

return $max(Rank)$

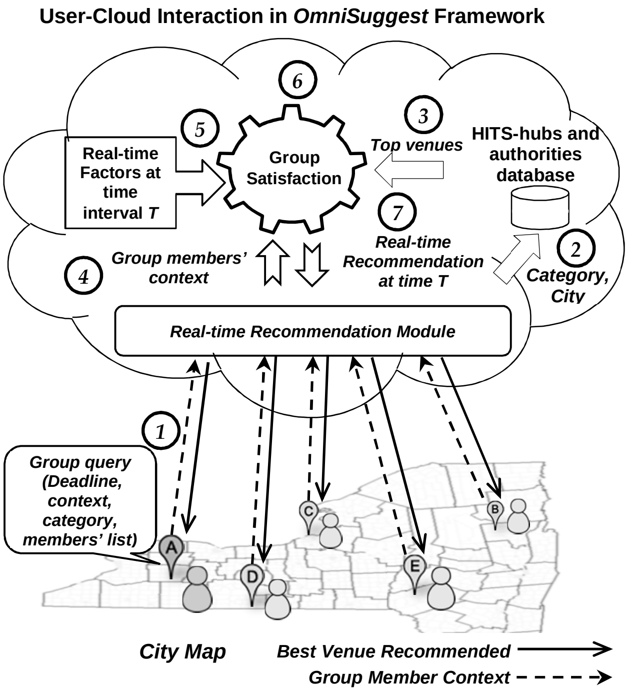


Fig. 8.5. A step-wise procedure for group recommendations.

8.3.3.2. Proposed Group Recommendation Approach

Formally, we state the group recommendation problem as: *"Given a list of venues V in a geographic region R, and a given category C, recommend a venue to a group, such that, it maximizes the group's satisfaction, under each member's individual context."* Algorithm 8.2 presents our proposed approach that is further illustrated using Fig. 8.5

1. *Initializations (Line 1 and Line 2).* The algorithm takes as input the group query sent by the group leader, which includes the list of group members, deadline by which members must reach the venue of a given category, and the geographical region (Fig. 8.5, Step 1). Line 2 of the algorithm retrieves a set of top venues based on their pre-calculated HITS score from the database. The top venues retrieved are for the given category and geographic region (Fig. 8.5, Step 2 and Step 3).

2. *Real-time processing for each member (Line 3–Line 8).* In this step, the recommendation framework collects each group member's context, such as his/her current location, distance, and road condition leading to the top venues. In the next step, for each member, the approximate time in reaching the top venues is calculated (Line 3–Line 5). This approximate time is subtracted from the deadline ($\mathcal{T}$), and then multiplied with number of check-ins of a member *m* at a venue *v*, and authority score A_v of the venue *v* (Line 5 and Line 6). The multiplication result is stored in the matrix $[P]_{m\times v}$. If the parameter T_{mv} at Line 5 turns out to be lesser than 0, then the user will not be able to reach the venue within the deadline period, and that venue is dropped.

3. *Venue recommendation based on group satisfaction (Lines 9–Line 10).* In this step, venues are ranked by aggregating the values of matrix $[P]_{m\times v}$. The following aggregation functions are utilized [11]: *Average*, *Least Misery*, *Most Pleasure*, and *Approval Voting* (see Step 6, Fig. 8.5). At time interval *T*, a venue gets a top ranking, if it satisfies every group member. For example, if for a venue, most of the group members have higher ratings in the matrix $[P]_{m\times v}$, and we utilize the *Average* aggregation function, then the venue would be recommended as the best possible venue.

A venue that is placed on top of the recommended list at time *T*, may not stay at the same position at time interval $(T + \Delta t)$, where Δt is the increment in the initial time (T). Therefore, to ensure that the SLA is maintained, the *OmniSuggest* framework generates a second recommendation only in adverse conditions, when a few group members are unable to reach the recommended venue before the deadline. Such adverse conditions may arise due to road blocks and/or severe weathers. In this way, the framework combines venues' popularity, group members' individual preference to a venue, real-time conditions, and mutual consent of all of the group members to generate the venue recommendations.

8.3.4. Time Complexity

In this section, we present time complexity analysis of *OmniSuggest* framework. We compute the time complexity of offline pre-processing tasks, as well as online recommendation modules of Algorithm 8.1 and Algorithm 8.2.

8.3.4.1. Offline Time Complexity of HITS Method

The *OmniSuggest* framework utilizes the HITS approach to rank the popular venues and experienced users for each category in a geographical region. The time complexity of HITS method is $O(m \times (h'^2 + v^2))$, where the parameter m is the number of iterations required by the HITS method to converge, h' represent the total number of users in a region, and v is the number of venues in a category. For a total of l categories, the time complexity is $O(l \times m \times (h'^2 + v^2))$. Moreover, the computations of similarity graphs among experienced users h, for each category, takes $O(l \times h^2)$. Therefore, the total time complexity for offline pre-processing is $O(l \times ((m \times (h'^2 + v^2)) + h^2))$. Here, as we have $h' \gg h$, so we compute the complexity as $O(l \times m \times (h'^2 + v^2))$.

8.3.4.2. Time Complexity Analysis of Algorithm 8.1

The Line 3 of Algorithm 8.1 computes the active user's similarity with a set of experienced users. The complexity of the similarity function for v venues is $O(v)$. Therefore, total time complexity of Line 3 is $O(h \times v)$. The worst-case scenario is that an active user's similarity evaluates to be greater than zero for all h experts. The creation of k ants (Line 4) takes $O(k)$. The Line 25 updates pheromone trail with complexity $O(h)$. The Line 7 takes a time complexity of $O(h \times log\ h)$ to sort the h experts (neighbors) with respect to their pheromones trails. In the worst case, Line 8–Line 13, number of iterations is h, and the Line 7–Line 24 are also iterated for h times, unless the stopping criterion is met (Line 26). Time complexity for both the pheromone update phase (Line 25) and aggregate function (Line 30) is $O(h)$. We conglomerate the overall time complexity of Algorithm 8.1 to be as $O(h \times (v + h \times \log h))$. The number of ants generated at Line 4 equals to the number of neighbors. Therefore, we use $k = h$ in calculating time complexity. The complexity of sequential execution of Algorithm 8.1, without considering the parallel execution of ants, is given as: $O(h \times v + 3h^2 + h^2 \times \log h + 2h) = O(h \times v + h^2 \times \log h) = O(h \times (v + h \times \log h))$. By executing the ants in parallel, the time complexity of the Algorithm 8.1 is further reduced to $O(h \times (v + \log h))$.

8.3.4.3. Time Complexity Analysis of Algorithm 8.2

The time complexity of Line 3 - Line 8 is $O(g \times v)$, where the parameter g is the number of group members, and v is the number of venues. The complexity of Line 9–Line 10 is $O(v)$. The overall complexity of Algorithm 8.2 in the sequential case is $O(g \times v)$. In parallel case, the complexity of Algorithm 8.2 is reduced to $O(v)$.

From the above analysis, we can deduce that significant speed up is achieved by the underlying cloud infrastructure that facilitates elastic parallel executions. Therefore, when the user volume is high, greater number of cloud nodes can be deployed to scale-up the performance, and conversely, scale-down.

8.4. Performance Evaluation

In this section, we perform the experimental evaluation of the proposed cloud based *OmniSuggest* framework. For the comparison purposes, we selected the following existing recommendation techniques (defined in the next subsection): **(a)** Popularity based ranking [8.9], **(b)** Social-based ranking [8.9], **(c)** User-based collaborative filtering (UBCF) [8.1], [8.2], and **(d)** SVD matrix factorization [8.2].

8.4.1. Related Recommendation Techniques

- *Popularity based ranking* approach assigns popularity to venues depending on the number of heck-ins. The rank $\hat{r}_j$ for a venue is computed as $\hat{r}_x = \sum_{i \in U} v_{ix}$, where v_{ix} is number of visits of a user *i* to a venue *x*.
- *Social-based Ranking* computes the venue popularity ranking by utilizing social network profiles of users. For a given user, the popularity of a particular venue depends on the number of check-ins performed by friends [8.9].
- *User-based Collaborative Filtering (UBCF)* methods, such as *k*-Nearest Neighbor (*k*-NN) [8.2] measure the similarity within the users' profiles to find the extent to which users visit the same venues. Based on the similarities, *k*-NN set of a given user is computed. The nearest neighbor set is utilized to generate rating for a venue by using the following relationship $\hat{r}_{a,j} = \bar{r}_a + \frac{\sum_{x \in U} sim(a,x)(v_{xj} - \bar{r}_b)}{\sum_{x \in U} sim(a,x)}$, where $\bar{r}_a$ is the mean number of check-ins by a user a.
- *Singular Value Decomposition (SVD)* matrix factorization method [8.2], maps users and venues to a joint latent factor space of dimensionality *f*. A user *u* is associated to a row vector represented by $p_u \in \mathbb{R}^f$, and a venue *v* is associated with a column vector given by $q_u \in \mathbb{R}^f$. A user's estimated rank for a venue *v* is represented as $\hat{r}_{u,v} = q_v{}^T \times p_u$. To estimate the values of q_v and p_u, the regularized- squared-error is minimized in the system given by $E = min \sum_{(u,v) \in K} \left(r_{u,v} - q_v^T p_u\right)^2 + \lambda(|q_u|^2 + |p_v|^2)$ where $r_{u,v}$ is the rating of user *u* for venue *v* and λ controls the regularization extent, and is determined by cross validation.

8.4.2. Aggregation Strategies

We selected the following aggregation strategies presented in [8.11] for the group recommendations: **(a)** *Least Misery,* **(b)** *Most Pleasure,* **(c)** *Average Satisfaction,* and **(d)** *Approval Voting*. The *least misery* strategy selects the rating of a user who has the minimum ratings for venue *v* within the group. The group rating GR_v of venue *v* is calculated as $GR_v = min(r_{u,v})$. The *most pleasure* strategy selects the rating of a user who has the maximum ratings for venue *v* within the group. The group rating GR_v of venue *v* under *most pleasure* is represented as $GR_v = max(r_{u,v})$. The *average satisfaction* strategy computes the group rating GR_v of venue *v* by taking the average of the ratings ($r_{u,v}$) of the group members. The formula for group average satisfaction is given as $GR_v = \frac{1}{n} \times \sum_{u=1}^{n}(r_{u,v})$. The *approval voting* rates a venue based on counting the group members who have ratings above a certain threshold. For example, counting the number of group members that can reach a venue before the deadline

The aggregation strategies are applied with selected schemes, such as *SVD*, *POPULAR*, and the *OmniSuggest* framework to aggregate the rankings of venues for the group (Fig. 8.6(d), (e), and (f)). The venue selected for recommendation to a group in any of the above mentioned aggregation strategies is the maximum value of GR_v among all the available venues, given as $GR = max(GR_v)$.

8.4.3. Results

In this section, we present the evaluation results of the proposed cloud based *OmniSuggest* framework. As the *OmniSuggest* framework takes into account the real-world time varying parameters, the traditional evaluation mechanisms, such as Min-Cut and Map-Reduce cannot be applied to measure the performance. Therefore, instead of finding out the convergence times of the proposed algorithms, we emphasize on the fact that the suitability of SLA is more critical than other intrinsic issues that are important in supercomputing, high performance computing, and data intensive computing environments [8.18]. In the *OmniSuggest* framework, we have based the SLA on the satisfaction of users with the recommended venues, instead of the response time as in traditional frameworks. It is noteworthy to mention that because *OmniSuggest* must respond to real-time traffic information, its response time would be superior to other systems, due to the parallelization in the processes. However, to quantify such a measure is not meaningful for the problem at hand.

We have carried out experiments on our internal Ubuntu cloud setup running on 96 core Supermicro SuperServer SYS-7047GR-TRF systems. The data flow process between the end users and cloud is depicted in Fig. 8.5. To perform the evaluations we used the *Foursquare* dataset [8.4] that consists of 425,680 tips provided by 49,027 users for 206,416 venues in New York and 327,431 tips given by 38,134 users in Los Angeles. The users' check-in history is split into two portions: **(a)** training set (80% of the records) and **(b)** testing set (20% of the records). In the following subsections, we discuss the evaluation results for single user case and group recommendations.

8.4.3.1. Evaluation of Single User Recommendation

The Algorithm 8.1 reports the best performance for $\rho = 0.01$ and $t_{max} = 100$ that are determined empirically through numerous runs on varying datasets. The value $t_{max} = 100$ is large enough as data is already refined in the preprocessing phase. To evaluate the single user recommendation effectiveness, we use the following performance metrics [8.1]: **(a)** Precision, **(b)** Recall, and **(c)** F-measure. Precision is defined as the ratio of correct recommendations (true positives (*tp*)) to the total number of recommendations (*tp* + false positives (*fp*)). The correct recommendations count is computed as follows. For a given user, the ratings of randomly selected venues are set blank. Thereafter, the recommendation framework generates top-*N* venues for the user. The correct recommendations are the number of venues appearing as the intersection of the aforementioned top-*N* venues, and the venues that were set blank for evaluation. Precision gives the average quality of the individual recommendations, and can be represented as:

$$Precision = \frac{tp}{tp + fp}. \tag{8.10}$$

Recall is defined as a ratio of hit set size to the total size of test set, and is the measure of the recommendation coverage by a recommendation system, given as:

$$Recall = \frac{tp}{tp + fn}. \tag{8.11}$$

F-measure is the harmonic mean of precision and recall

$$F\text{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{8.12}$$

As reflected in Fig. 8.6(a) and Fig. 8.6(b), the *OmniSuggest* framework achieves the best performance in terms of precision and recall, compared to the rest of the schemes (each of the plot shows the average of 100 random runs). This is because the *OmniSuggest* framework provides a more effective solution towards the data sparsity problem by augmenting similarity computations with conditional probabilities and bifurcating the check-ins data into sub-categories. The reduction in data sparseness results in an increased recommendation precision. The well-known collaborative filtering technique, such as *SVD* and *UBCF* [8.1], [8.2], indicated low performance in terms of precision and recall due to higher data sparseness. Moreover, *UBCF* is not shown in plots as it failed to produce any results on the highly sparse dataset of *Foursquare* considered in our experiments. The popularity-based approaches, such as *SOCIAL* and *POPULAR* performed better than the collaborative filtering techniques. The reason is that popularity-based approaches do not utilize similarity computations in their models. Therefore, these approaches are not significantly affected by data sparsity problems. The recall of *OmniSuggest* framework is the highest for *N=20.* This indicates that the framework provides a greater coverage in terms of recommendations. However, increase in coverage comes at the cost of lower precision values. The tradeoff between precision and recall is depicted in Fig. 6c. Compared to other schemes, the cloud based *OmniSuggest* framework indicates better performance in terms of the F-measure. The improved F-measure performance is due to the higher values of precision and recall at *N=10*. The performance of *RANDOM* remains low for all the aforementioned metrics. This is because, *RANDOM* simply shuffles the candidate set of unvisited locations for each user, without performing similarity computations.

8.4.3.2. Evaluation of Group Recommendation

The group recommendation is evaluated by employing the following aggregation strategies, as elaborated in [8.11]: **(a)** *Average,* **(b)** *Least Misery,* **(c)** *Most Pleasure,* and **(d)** *Approval Voting*. To imitate the real-world physical factors, we generated a random set of parameters for speed, distance, and road conditions. To ensure fairness in results, all of the recommendation models are evaluated with the same set of random parameters. (Each of the plot shows the average of 100 runs.) The traditional performance metrics, such as precision and recall, cannot be utilized for group based recommendations. This is because, the groups are created on the fly and groups may have different number of users, which makes it impossible to store the group specific history in a database. Therefore, we utilized a performance metric *global satisfaction (gs*) [8.17] to evaluate the group recommendations:

$$gs(G) = \overline{S} - \sigma_s, \tag{8.13}$$

where G represents the group, $0 \leq gs(G) \leq 1$ is the global satisfaction for all of the group members, $\overline{S}$ is the mean, and σ_s represents the standard deviation of satisfaction. The global satisfaction, *gs*, provides a measure of similarity within the satisfaction level of all of the group members. An individual's satisfaction level S in the group is defined by the following formula [8.17]:

$$S(u,G) = \begin{cases} 1.0, \text{if } esl(GR, list_u) \leq 03; \\ 0.9, \text{if } esl(GR, list_u) \leq 04; \\ 0.8, \text{if } esl(GR, list_u) \leq 06; \\ 0.6, \text{if } esl(GR, list_u) \leq 08; \\ 0.4, \text{if } esl(GR, list_u) \leq 10; \\ 0.2, \text{if } esl(GR, list_u) \leq 12; \\ 0.0, \text{if } esl(GR, list_u) > 12. \end{cases} \tag{8.14}$$

In above equation, the parameter *GR* represents the recommendation generated for the whole group, and $list_u$ is the recommendation list for each of the individual user. The Expected Search Length (esl) function maps the satisfaction level of an individual to the recommendation generated for the whole group. The function assigns a scale within range $[0\ 1]$ to the index of a group recommended venue (GR) in an individual member's venues' list ($list_u$). The greater value returned by esl function means a group recommended venue is appearing amongst the preferred venues' list of a given member.

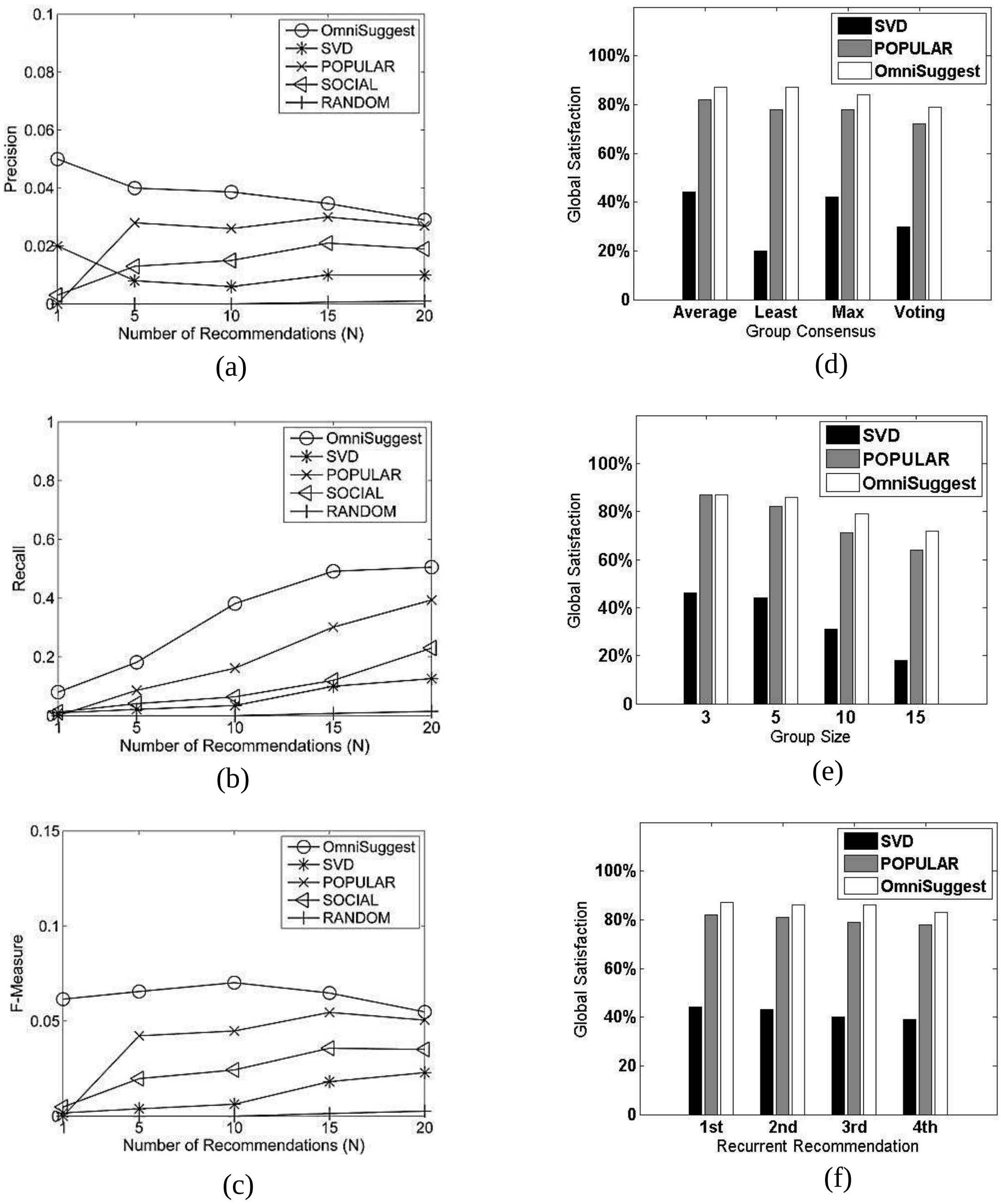


Fig. 8.6. Performance evaluation results: (a) precision, (b) recall, (c) F-measure, (d) group consensus effects, (e) group size effects, and (f) effect of recurrent recommendations on global satisfaction.

Fig. 8.6(d) depicts global satisfaction results for a group of five members. The improved performance of proposed framework for all aggregation strategies is because of the measures taken by the *OmniSuggest* framework to handle the data sparseness. The *SVD* scheme being sensitive to data sparseness, does not exhibit uniformity in the personal opinions of group members. Therefore, the increased standard deviation value decreases the overall satisfaction score for *SVD* in all of the aggregation strategies. The performance of *OmniSuggest* and *POPULAR* is almost similar for *Average*, *Least Misery*, and *Most Pleasure* aggregation strategies. This is due to the fact that these schemes are less sensitive to data sparseness, and indicate uniformity in the satisfaction level of all group members.

Fig. 8.6(e) reflects the global satisfaction by varying the group size. For all of the recommendation approaches, the *Average* function is selected as an aggregation strategy. The increase in group size resulted into a decrease in global satisfaction for the three recommendation approaches. Moreover, increase in the number of group members also increases the deviation in the satisfaction of individual members, which results in overall decrease of global satisfaction in (13).

The system generates new recommendations only when significant change in group members' context occurs (such as road blocks). It can be observed from Fig. 6f that a recurrent recommendation has an insignificant effect on the global satisfaction. The reason is that on all occasions, the recommended venue is the one that encapsulates the users' mutual consent based on their current context. However, it is noteworthy to mention that the change in the recommendation may cause a negative effect on the mood of the group members, as they have already travelled some distance towards the previously recommended venue. Therefore, every new recommendation generated by the system may decrease the overall satisfaction by a factor, given as below.

$$S_j(u,G) = \delta^j \times S_c\ (u,G), \tag{8.15}$$

where $S_c\ (u,G)$ represents a user's satisfaction for a new venue recommended by the system, and δ^j is scaling factor that depicts the decay in satisfaction over a period of time. The parameter j

indicates the number of times recurrent recommendations are made. As depicted in Fig. 8.7, greater the value of j, lower the satisfaction level for the group in all the recommendation approaches.

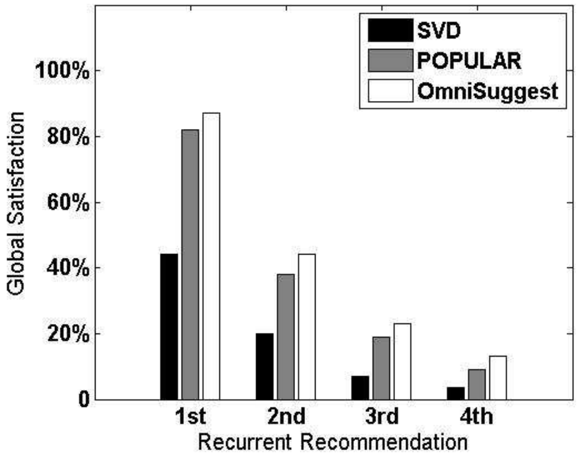


Fig. 8.7. Global satisfaction with the mood impact on recurrent recommendation.

To summarize the results, it is evident that our cloud based *OmniSuggest* framework demonstrated an overall better performance, as the proposed framework has more efficient mechanism of handling data sparsity problem. Moreover, in the case of the group recommendation, the cloud based *Omnisuggest* framework provides a higher level of satisfaction in terms of recommended venues to the group members.

8.5. References

[8.1] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, “Recommender systems survey,” *Knowledge-Based Systems*, vol. 46, pp. 109-132, July 2013.

[8.2] L. Lü, M. Medo, C.H. Yeung, Y. Zhang, Z. Zhang, T. Zhou, “Recommender systems,” *Physics Rezports*, vol. 519, no. 1, pp. 1–49, 2012.

[8.3] J. Levandoski, M. Sarwat, A. Eldawy, and M. Mokbel, “Lars: A location-aware recommender system,” In *IEEE 28th International Conference on Data Engineering (ICDE)*, pp. 450-461, 2012.

[8.4] J. Bao, Y. Zheng, M.F. Mokbel, “Location-based and preference-aware recommendation using sparse geo-social networking data,” In *Proc. 20th International Conference on Advances in Geographic Information Systems*, ACM New York, pp.199-208, 2012.

[8.5] M. Ye, P. Yin, and W. Lee, “Location recommendation for location-based social networks,” In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, pp. 458-461, 2010.

[8.6] C. Chow, J. BAO, and M. Mokbel, “Towards Location-Based Social Networking Services,” In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, ACM, pp. 31-38, 2010.

[8.7] Y. Doytsher, B. Galon, and Y. Kanza, “Storing routes in socio-spatial networks and supporting social-based route recommendation,” In *Proc. 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, ACM, pp. 49-56, 2011.

[8.8] K. Chang, L. Wei, M. Yeh, and W. Peng, “Discovering personalized routes from trajectories,” In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, ACM, pp. 33-40, 2011.

[8.9] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo, “A Random Walk Around the City: New Venue Recommendation in Location-Based Social Networks,” In *Proc. 2012 International Confernece on Social Computing (SocialCom)*, IEEE, pp.144-153, 2012.

[8.10] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*, Cambridge University Press, 2010.

[8.11] J. Masthoff, "Group recommender systems: Combining individual models." *Recommender Systems Handbook*, Springer US, 2011, pp. 677-702.

[8.12] L. Wei, Y. Zheng, and W. Peng, "Constructing popular routes from uncertain trajectories," In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 195-203, 2012.

[8.13] N. Tziritas, C.-Z. Xu, J. Hong, and S.U. Khan, "An Optimal Fully Distributed Algorithm to Minimize the Resource Consumption of Cloud Applications," In *18th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, IEEE, pp. 61-68, December 2012.

[8.14] D. Kliazovich, P. Bouvry, and S.U. Khan, "Simulation and Performance Analysis of Data Intensive and Workload Intensive Cloud Computing Data Centers," *Optical Interconnects for Future Data Center Networks*, C. Kachris, K. Bergman, and I. Tomkos, Eds., Springer-Verlag, New York, USA, ISBN: 978-1-4614-4629-3, Chapter 4.

[8.15] J. Li, P. Roy, S.U. Khan, L. Wang, and Y. Bai, "Data Mining Using Clouds: An Experimental Implementation of Apriori over MapReduce," In *12th International Conference on Scalable Computing and Communications (ScalCom*), Changzhou, China, December 2012.

[8.16] P. Bedi and R. Sharma, "Trust based recommender system using ant colony for trust computation," *Expert Systems with Applications: An International Journal archive*, vol. 39, no. 1, pp. 1183-1190, 2012.

[8.17] L. Quijano-Sanchez, J.A. Recio-Garcia, B. Diaz-Agudo, and G. Jimenez-Diaz, "Social factors in group recommender systems," *ACM Transactions on Intelligent Systems and Technology (TIST)* , vol 4, no. 1, 2013.

[8.18] L. Wang and S. U. Khan, “Review of Performance Metrics for Green Data Centers: A Taxonomy Study,” *Journal of Supercomputing*, vol. 63, no. 3, pp. 639-656, 2013.

[8.19] H.S. Chiang, T.C. Huang, “User-adapted travel planning system for personalized schedule recommendation,” *Information Fusion*, 2013.

[8.20] J. Oh, O. Jeong, and E. Lee, “Collective Intelligence Based Place Recommendation System,” In *Advanced Infocomm Technology*, Springer Berlin Heidelberg, pp. 169-176, 2013.

[8.21] Y. Zheng, L. Zhang, X. Xie, and W.Y. Ma, “Mining interesting locations and travel sequences from gps trajectories,” In *Proceedings of the 18th international conference on World wide web*, ACM, pp. 791-800, 2009.

9. CONCLUSION AND FUTURE WORK

This dissertation contributes to the development of a novel suite of protocols for resource efficient message dissemination in diverse mobile networks. We proposed six routing protocols for network environments that span from mostly disconnected DTNs and partially connected opportunistic mobile ad hoc networks, to mostly connected cellular networks. Moreover, we also present a novel application to generate optimal recommendations for a group of mobile users in a fully connected environment of mobile social networks. Designing protocols for wireless networks with high mobility and frequent disruptions is challenging because of the uncertainties involved in topology, connectivity, and channel conditions. By efficiently exploiting the connection opportunities among mobile nodes, the proposed schemes achieved better performance in terms of message delivery ratio and overhead. The proposed research and simulation results are presented in six separate chapters, and a brief summary of contributions is provided in the following subsection.

9.1. Summary of Contributions

In Chapter 3, we presented a detailed simulation and analysis of ten popular DTN routing protocols. Selecting the best protocol to be used in a given environment remains a difficult task, as comparisons are often clouded by different superfluous assumptions in the original design of a protocol. The main purpose of this chapter was to study and benchmark protocols on a unified platform with varying network parameters so as to fully understand capabilities and limitations of the protocols. The protocols were thoroughly evaluated using synthetic mobility environment as well as real-world connectivity traces by using the following parameters: (a) network size, (b) buffer size, (c) message rate, (d) message size, and (e) bandwidth. The results indicated that the protocols that utilized additional network information to route messages show better performance than the protocols that mostly relied on flooding based techniques. However, increasing the size of metadata and protocol complexity also increases computational cost and buffer requirements. It is further observed that the performance of DTN routing protocols is greatly affected by mobility pattern of nodes and message queue management policies. As an additional contribution, we proposed three routing techniques by introducing adaptability in the replication strategies of the three most cited DTN routing protocols. When compared with existing protocols, the proposed schemes indicated significantly improved performance. In future, we intend to expand the functionality of proposed techniques to make them a workable solution for providing opportunistic message transfer in a real DTN network environment consisting of heterogeneous nodes.

Chapter 4 presented a message routing protocol Forecast and Relay (FAR) for challenging environments of the OMNs. The novelty of the scheme lies in exploiting the forecasting techniques on the temporal data of past meeting qualities to perform future contact predictions. When compared with existing message delivery schemes, such as *PRoPHET*, *Epidemic*, *Random*, and *Wave* the proposed protocol showed significant performance improvement. Moreover, the evaluation results with real-world connectivity traces and large-scale synthetic mobility suggest that the *FAR* protocol to be an ideal content delivery scheme for diverse opportunistic and delay-tolerant application scenarios.

Chapter 5 exploits the human mobility behavior to develop a Check Point (CP) based architecture, in which the CPs are deployed on locations where human meetings are more frequent and each CP is covering a specific geographic location in the city of Fargo, ND, USA. The messages are relayed among CPs through buses following fixed schedules. The simulation results indicate that by installing the CPs on locations with higher human interactions increase the predictability of finding the message destination. Therefore, the CP based approach improves message delivery ratio, decreases buffer utilization of nodes and message delivery time.

In Chapter 6, we examined various challenges faced by the network when nodes are willing to participate in opportunistic data sharing and storage applications, to form on-the-fly data centers. We addressed replica placement as one of the major challenges in ad hoc based data storage. In these networks, the decision of where to replicate data must trade off the cost of accessing data. The data access cost can be reduced by replications of data items, but with additional cost of storing and updating the replicas. These costs have severe implications in ad hoc networks because mobile hosts have limited resources (energy, storage, and processing power). Therefore, efficient and effective replication schemes strongly depend on how many replicas to be placed in the system, and more importantly where. We performed a comparative study of some of the well-known data replication schemes for MANETs, and discussed various pros and cons of the studied schemes. We observed that data replication is quite challenging in DTN like environments due to the non-existence of end-to-end communication paths. Unlike MANETs, the lack of end-to-end connectivity in DTNs prevents the global network information propagation. We formulated the data replication problem in DTNs and proposed a utility based replication scheme. The aforementioned utility value was based on two things: (a) probability that the node will be able to deliver message before life time expiry, and (b) probability that node will stay in contact with message's destination long enough to compensate message transfer time. Our results from synthetic mobility as well as real-world traces indicated that the proposed scheme produced the minimum network cost, and maximum delivery ratio. As a future work, we intend to explore numerous opportunistic message storage and sharing applications for bus-based DTNs and vehicular ad hoc networks (VANETs).

Chapter 7 presented a message routing protocol Adaptive Prognostic Scheme (APS) for challenging DTN environments. The novelty of the proposed protocol lies in exploiting Auto Regressive Integrated Moving Average (ARIMA) model on temporal data of past contacts to perform optimal message routing. When compared with the existing message delivery services, the *APS* protocol showed improved performance. Moreover, the evaluation results with two large-scale city based DTN scenarios suggest that the *APS* protocol appears to be an ideal content delivery scheme for numerous DTN applications.

In Chapter 8, we presented a multifold contribution by devising cloud based solutions for the venue recommendation problem in mobile social networks for a single user and/or a group of friends. The novelty and significance of this work was the integration of knowledge engineering techniques, such as HITS method, Ant colony optimization, and collaborative filtering on a cloud infrastructure to generate optimal set of recommendations. Different from the previous works, the proposed *OmniSuggest* framework not only took into account the collective opinions of the experienced users, but also considers the effect of dynamic real-world physical factors, such as a person's distance from venues, speed, weather conditions, and travel conditions. The scalability issues were addressed by proposing a cloud-based architecture that allocated data and computational load on geographically distributed cloud nodes. Data sparseness issues were resolved by augmenting similarity computations with conditional probabilities and further refining the data storage by bifurcating data into multiple levels of predefined categories. In this way, the *OmniSuggest* framework always had a precompiled set of experienced users for any category and was able to recommend best venues for a new user at finer granularity. The evaluation results with real-world *Foursquare* dataset indicated the improved performance of the proposed *OmniSuggest* framework than many of the existing schemes. Our study revealed that real-world physical conditions have a significant effect on the final recommendations, when combined with users' context.

9.2. Future Work

In this section, we highlight some of the research directions that we intend to explore in future. First of all, we will devise message transfer schemes for the data in bulk. An example of such application is sensor networks deployed at experimental sites to extract environmental data. The massive volumes of data extracted by sensors needs to be transferred towards the laboratories situated in cities at far distances. We want to exploit the passing-by vehicles to voluntarily receive data from sensor sites, and deliver to the destinations. As the vehicular nodes are mobile, the contact durations are very limited, so data transfer must be made optimal by transferring maximum data during the limited contact period. We intend to utilize various network coding schemes, such as erasure coding and random coding to improve the throughput of bulk data transferred. We will also explore multicasting in mobile networks that transfer data to only a selected set of nodes, so that overhead can be further reduced. Moreover, we will like to focus on the trust and security issues in routing of messages in diverse mobile network environments as by nature, such networks are very open to different attacks and the routing of messages can easily fail due to these attacks. To extend our work on mobile social networks, we plan to further combine approaches from multiple disciplines, such as artificial neural networks, Bayesian networks, and machine learning techniques to devise solutions that efficiently handle the data sparseness, cold start, and scalability issues. Moreover, we intend to integrate the recommendation module with early disaster warning systems, such as information about tornados, landslides, tsunamis, and floods, which would help in generating recommendations closely depicting real-world scenarios.