12-2012

Towards Trustworthy, Efficient and Scalable Distributed Wireless Systems

Ze Li
Clemson University, zel@clemson.edu

Towards Trustworthy, Efficient and Scalable Distributed Wireless Systems

---

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Computer Engineering

---

by
Ze Li
December 2012

Accepted by:
Dr. Haiying Shen, Committee Chair
Dr. Kuang-Ching (KC) Wang
Dr. Melissa C. Smith
Dr. Amy Apon

Abstract

Advances in wireless technologies have enabled distributed mobile devices to connect with each other to form distributed wireless systems. Due to the absence of infrastructure, distributed wireless systems require node cooperation in multi-hop routing. However, the openness and decentralized nature of distributed wireless systems where each node labors under a resource constraint introduces three challenges: (1) cooperation incentives that effectively encourage nodes to offer services and thwart the intentions of selfish and malicious nodes, (2) cooperation incentives that are efficient to deploy, use and maintain, and (3) routing to efficiently deliver messages with less overhead and lower delay. While most previous cooperation incentive mechanisms rely on either a reputation system or a price system, neither provides sufficiently effective cooperation incentives nor efficient resource consumption. Also, previous routing algorithms are not sufficiently efficient in terms of routing overhead or delay.

In this research, we propose mechanisms to improve the trustworthiness, scalability, and efficiency of the distributed wireless systems. Regarding trustworthiness, we study previous cooperation incentives based on game theory models. We then propose an integrated system that combines a reputation system and a price system to leverage the advantages of both methods to provide trustworthy services. Analytical and simulation results show higher performance for the integrated system compared to the other two systems in terms of the effectiveness of the cooperation incentives and detection of selfish nodes.

Regarding scalability in a large-scale system, we propose a hierarchical Account-aided Reputation Management system (ARM) to efficiently and effectively provide cooperation incentives with small overhead. To globally collect all node reputation information to accurately calculate node reputation information and detect abnormal reputation information with low overhead, ARM builds a hierarchical locality-aware Distributed Hash Table (DHT) infrastructure for the efficient and integrated operation of both reputation systems and price systems. Based on the DHT infrastructure, ARM can reduce the reputation management overhead in reputation and price systems. We also design a distributed reputation manager auditing protocol to detect a malicious reputation manager. The experimental results show that ARM can detect the uncooperative nodes that gain fraudulent benefits while still being considered as trustworthy in previous reputation and price systems. Also, it can effectively identify misreported, falsified, and conspiratorial information, providing accurate node reputations that truly reflect node behaviors.

Regarding an efficient distributed system, we propose a social network and duration utility-based distributed multi-copy routing protocol for delay tolerant networks based on the ARM system. The routing protocol fully exploits node movement patterns in the social network to increase delivery throughput and decrease delivery delay while generating low overhead. The simulation results show that the proposed routing protocol outperforms the epidemic routing and spray and wait routing in terms of higher message delivery throughput, lower message delivery delay, lower message delivery overhead, and higher packet delivery success rate. The three components proposed in this dissertation research improve the trustworthiness, scalability, and efficiency of distributed wireless systems to meet the requirements of diversified distributed wireless applications.

Acknowledgments

I would like to thank my advisor Dr. Haiying Shen for her unending patience and guidance. I am also deeply grateful for the opportunity to have had Dr. Melissa Smith, Dr. Kung-Ching Wang, and Dr. Amy Apon as my committee members at Clemson. Their professionalism and enthusiasm for research inspired me to pursue my Ph.D.

My experience at Clemson University has been joyful and enlightening and I am grateful to the members of the Pervasive Computing group for their insight, constructive criticism, and friendship.

Finally, I must thank my friends and family for their encouragement and support. In particular, I am grateful to my parents for their love and for instilling in me a deep sense of academic pride.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Background

Background Tremendous advances in wireless technologies enable distributed mobile devices (e.g., cell phones, laptops and smartphones) to connect with each other without a fixed infrastructure to form distributed wireless systems such as mobile ad hoc networks (MANETs) and delay tolerant networks (DTNs). Distributed wireless systems are playing an increasingly important role in areas such as commerce, emergency services, military, education and entertainment. Compared to a centralized system that is prone to problems such as congestion, single point of failure, a distributed wireless system can reduce the computation and bandwidth burdens on a centralized system and increase the reliability by taking advantage of the distributed resources in the autonomous peers. The number of mobile device users has been increasing rapidly. Studies show that the number of wireless Internet users has tripled world-wide in the last three years, and the number of smartphone users in the US has increased from 7.4 million in 2003, to 69.2 million in 2006, to 190 million in

2010 [5], and is expected to reach 300 million by 2013 [1]. The computational capability of wireless devices is increasingly powerful as well. For example, the dual-core A5 processor in the iPhone4S runs at 800MHz with a storage capacity up to 64GB [3]. The iPhone4S is also enabled with a cellular interface (i.e. UMTS/HSDPA/HSUPA and GSM/EDGE) and WiFi interface (i.e., 802.11b/g/n). Such high-capability devices can support many envisioned distributed wireless system applications, such as data sharing [8], road traffic monitoring [84], emergency assistance services [39], and multimedia data transmission [36].

Due to the absence of infrastructure in distributed wireless systems, the systems require the cooperation of the nodes in the system to collaboratively conduct a task. For example, MANET requires the cooperation of every node in the path for successful packet transmission from a source node to a destination node. However, distributed wireless systems are particularly vulnerable to selfish node behaviors due to the individualized nature of nodes. Each node labors under an energy constraint, and selfish nodes tend not to forward packets to conserve their own resources for their own. It has been proven that the presence of only a few selfish nodes can dramatically degrade the throughput of an entire system [68]. What's worse, identifying and punishing selfish nodes will decrease the throughput of cooperative nodes because of the longer transmission path or even lead to complete network disconnection [45]. Therefore, rather than simply punishing selfish nodes, detecting selfish nodes in a packet transmission and further encouraging nodes to be cooperative is critical to ensuring the proper functionality of distributed networks. Limited resources are also an inherent problem in MANETs. The recent increasing growth of multimedia applications (e.g., video transmission) further imposes higher requirements of resource-efficiency. As a result, the openness and decentralized nature of MANETs with limited resources introduces three challenges: (1) effective cooperation incentives that encourage nodes to offer services and thwart the intentions of selfish and malicious nodes, (2) efficient cooperation incentives that are resource-efficient in use and maintenance, and (3) efficient routing to deliver messages

with less overhead and lower delay.

Previous Approaches To tackle the challenges for trustworthy node communication, most previous cooperation incentive mechanisms rely on either a reputation system [49, 74, 76, 82, 78, 69, 83, 14] or a price system [66, 56, 57, 85, 43, 54, 37]. However, neither system provides sufficiently effective cooperation incentives nor efficient resource consumption. Insufficient effectiveness means that the mechanisms cannot very accurately evaluate node reputation or prevent nodes from gaining unfair benefits while still being selfish. The accuracy of reputation evaluation can be adversely affected by false information including falsified [1], conspiratorial [2] and misreported information [3]. Insufficient efficiency means that the mechanisms exacerbate the resource-efficiency problem in large-scale wireless systems by consuming already scarce resources.

In most current reputation systems [76, 82, 78, 69, 11, 14, 81, 70, 63], each node periodically exchanges local reputation information with its neighbors and aggregates it to yield others' reputation values (R), which are referenced for forwarder selection in routing. A node with a reputation below a predefined threshold (T_R) is considered selfish, and otherwise is considered trustworthy. Selfish nodes' forwarding requests are always rejected by other nodes. However, such reputation systems have several deficiencies. First, reputation calculation based on local information, possibly including false information, may result in an insufficiently accurate reputation evaluation to truly reflect node trustworthiness. Second, they provide equal treatment to trustworthy nodes with $R > T_R$ and to selfish nodes with $R < T_R$. Thus, they cannot reward trustworthy nodes or punish selfish nodes differently. Also, nodes may first be cooperative and then later uncooperative while maintaining $R \geq T_R$

[1] Falsified information is reported by a dishonest node to deliberately increase or decrease others' reputations.

[2] Conspiratorial information is generated by colluders that report high reputations for each other to raise their own reputations, and report low reputations for others to decrease their reputations.

[3] Misreported information means low reputations for cooperative nodes that cannot offer high-quality transmission service due to adverse network conditions such as background interference.

to avoid punishment. Thus, solely relying on reputation systems cannot effectively deter misbehavior since clever selfish nodes can manipulate the policies to gain an unfair advantage while maintaining a trustworthy status. Third, because of node mobility, local reputation exchanges cannot give a node a reasonable understanding of a new neighbor that moves into its range. Fourth, they lack efficient mechanisms to collect and propagate reputation information. Exchanging information periodically, storing redundant reputation values in nodes, and broadcasting reputation queries consume significant resources.

Price systems [66, 56, 57, 85, 37] treat packet forwarding as a service transaction, and introduce virtual credits (or virtual money, stamps, points) for the transactions. In these systems, nodes forward packets for others to earn credits for their own packet transmission. However, the price systems have a number of inherent problems. First, the systems fail to provide a mechanism to monitor the service quality offered by a node. Second, they lack an effective method to punish selfish and wealthy nodes (e.g., nodes requiring few services) that occasionally drop others' packets. Also, cooperative nodes within a low-traffic region receive few forwarding requests, and thus may not earn sufficient credits for their needs and thus may drop some packets. Third, the circulation of credits in the network requires a fair amount of computation and storage resources and increases traffic overhead. Fourth, the implementation of credits and virtual banks adds complexity with a high requirement on transmission security. For example, since credits are stored at the head of a packet that is transmitted through several nodes, how to prevent the credits from being stolen becomes a problem.

While directly combining a reputation system and a price system can foster cooperation incentives to a certain extent, it cannot easily resolve problems in the individual systems. Even worse, it generates doubled system overheads, making the problem of resource-efficiency and scalability even more severe. Thus, a formidable challenge to be addressed in this research involves how to efficiently combine and coordinate the two systems to avoid the

problems in the individual systems, ensuring they can be exploited to their fullest capacities in achieving highly effective and efficient cooperation incentives.

Concurrently, even if such a trustworthy distributed system is maintained, the multicopy-based routing mechanisms [75, 90, 94, 97] generally used in current distributed wireless systems generate significant system overhead, which quickly drains the energy of the mobile devices and prevents the whole system from effectively conducting distributed tasks. Some research works in the literature proposed to use probabilistic single-copy routing [87, 47, 9, 27, 17, 71], in which messages are forwarded to mobile nodes that have higher probabilities of meeting the destination node as measured by the contact frequency utility. Although the single-copy methods save node resources and produce lower transmission overhead, they are likely to suffer from severe transmission delay if a suboptimal forwarding node (i.e., a node not in the shortest source-destination path) is chosen.

Goal This research aims to meet the trustworthy, scalable and efficient node communication requirements of diversified distributed wireless applications. To accomplish the goal of this research, we propose, design and implement three mechanisms as described below.

1.2 Methods of Study

This dissertation research introduces *novel mechanisms* to support a trustworthy, scalable and efficient distributed system as shown in Figure 1.1.

To ensure the *trustworthiness* of the distributed system and provide effective cooperation incentives that encourage nodes to offer services and thwart the intentions of selfish and malicious nodes, we propose an integrated reputation/price system, integrated system in short, that leverages the advantages of both the reputation and price systems and overcomes their individual disadvantages, making reciprocity the focal point. The design is based on our understanding regarding the underlying incentives and deficiencies of the reputation and

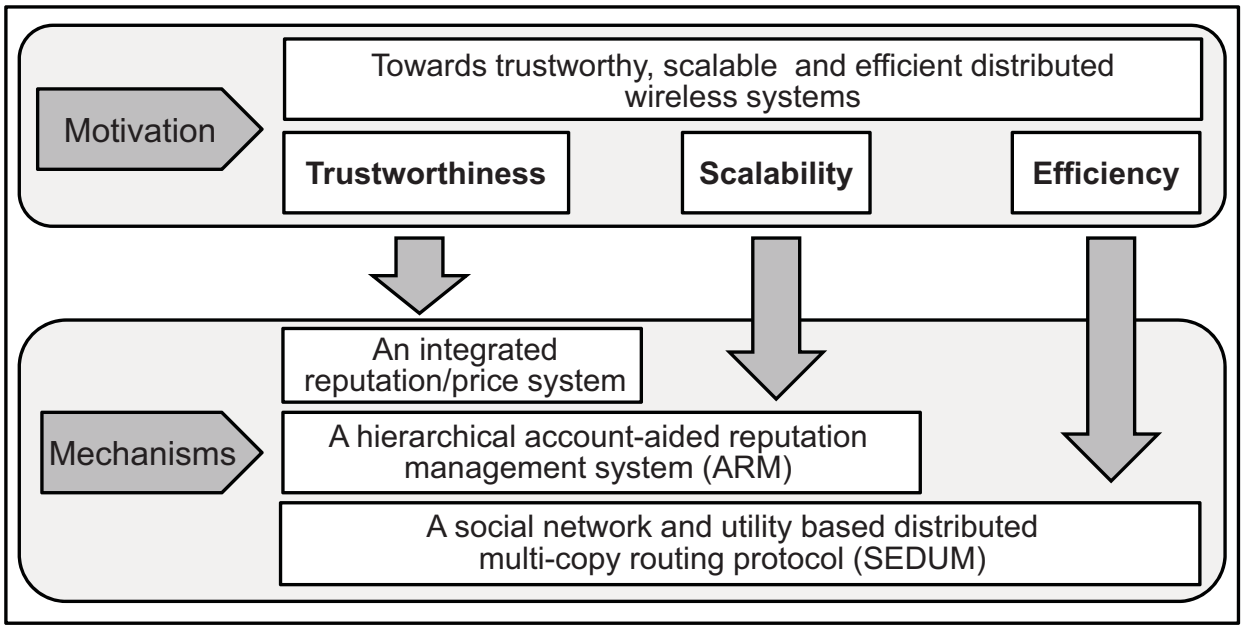


Figure 1.1: Motivation and mechanisms

price systems according to the game theory models. Specifically, the cooperative game theory model is used to explore how to form a rational coalition that can optimize the benefit of each node. We find that in the cooperative game model, each node earns the maximum benefit only when they form a grand coalition where all nodes in the system are cooperative. We also use a non-cooperative game model to investigate the best strategy for each node to maximize its benefit. We find that nodes can be uncooperative while still gaining benefits from both reputation and price systems. We then propose an integrated system that coordinately integrates a reputation system and a price system such that selfish nodes are unable to gain benefit from the system while being uncooperative. Our analysis of the integrated system shows that it can provide higher cooperation incentives than the reputation system and the price system alone, and more effectively detects selfish nodes.

To increase the *scalability* of the reputation management system as well as provide cooperation incentives that are resource-efficient in use and maintenance, we design a hierarchical Account-aided Reputation Management system (ARM). ARM selects low-mobility

and trustworthy nodes as reputation managers (managers in short), builds them into a locality-aware distributed hash table (DHT) [88] infrastructure, and coordinately integrates a reputation system and a price system through the infrastructure. DHT is well-known for high scalability, efficiency, and reliability, thus DHT supports scalable and efficient operations in ARM, by marshaling all node reputation and transaction information into one manager that calculates the reputation and increases/decreases credits in the account of the node accordingly. ARM can effectively deter selfish behaviors based on the integrated model studied in the integrated system. In addition, ARM can also effectively avoid reputation misreporting and collusion in reputation calculation for a large-scale system.

Using ARM as the basis for enabling trustworthy routing, we further investigate the design of an efficient routing protocol to increase the routing *efficiency* of the distributed wireless system. Noticing that in many DTN applications, such as mobile sensor networks [75], vehicular networks [39], and networks formed by mobile phone holders, the movement of mobile devices exhibits certain patterns in a social network because the device hosts (i.e., human or animals) normally have social movement routines [27]. Some nodes, such as home neighbors and colleagues, have a high probability of meeting (meet and contact are interchangeable in this dissertation) and staying close for a long time. This attribute of a movement pattern is called *colocation* [27] in a network. Some nodes, such as students on a campus, meet each other with high frequency but for a short meeting time. This attribute of the movement pattern is called *familiar stranger* in a social network [77]. Intuitively, familiar strangers normally have high contact frequency, but cannot guarantee the transmission of a large number of messages during a contact due to limited contact time. On the other hand, colocation nodes may have low contact frequency, but they have a long meeting time during each contact, where a large number of messages can be transmitted. For successful routing, we propose a routing protocol that can capture both the colocation and familiar stranger attributes in the social network. We first design a new metric, "duration utility",

captures both colocation and familiar stranger attributes. Based on the metric, we propose a multi-copy routing protocol to achieve a tradeoff between routing delay and overhead using an optimal tree replication algorithm and a Markov chain model. Further, we propose a buffer management mechanism to efficiently manage the messages in the buffer.

1.3 Research Contributions

The contributions of the dissertation research entail:

- Studying the cooperation incentives in a system without any incentive encouragement, a reputation system, and a price system for wireless distributed systems to ensure trustworthy communications (WSNS'08 [101] and ICCCN'09 [103], IEEE Transaction on Mobile Computing'11 [61]).
  - Designing several game theory models to analyze the incentives in current reputation systems, price systems and a system with no cooperation incentive strategy for cooperation encouragement
  - Designing an integrated system to leverage the advantages of both systems and to overcome their individual disadvantages, making reciprocity the focal point
  - Developing a game theory model to analyze the integrated system based on a cooperative game model and a non-cooperative game model
  - Presenting extensive evaluation results to justify the theoretical incentive models
- Proposing a hierarchical Account-aided Reputation Management system (ARM) to efficiently and effectively provide cooperation incentives (WiSP'08 [38], Infocom'11 [104]).
  - Studying the requirements to create a locality-aware DHT infrastructure in a MANET and proposing the construction and maintenance algorithms

  - Designing a reputation management protocol relying on the collected global information by DHT
  - Designing a distributed reputation manager auditing protocol to detect the malicious reputation manager
  - Designing a reputation-adaptive account management protocol to fairly treat nodes with different reputations and also to prevent nodes from gaining fraudulent benefits
  - Presenting extensive evaluation results to evaluate the performance of ARM

- Proposing a Social nEtwork and Duration Utility based distributed Multi-copy routing protocol (SEDUM) for a delay tolerant network that fully exploits node movement patterns in social networks to increase routing throughput and decrease routing delay (ICPP'08 [102], IEEE Transaction on Computers [62]).

  - Studying a *duration utility* based routing protocol that can provide optimal routing performance
  - Proposing a multi-copy routing protocol to achieve a tradeoff between routing delay and overhead based on an optimal tree replication algorithm and a Markov chain model
  - Proposing a buffer management mechanism to efficiently manage the messages in the buffer
  - Presenting extensive experiment results on social network and utility based distributed multi-copy routing protocol

1.4 Dissertation Organization

In the next chapter, we present an overview of the research areas of reputation systems and price systems geared towards mobile networks. We also present an overview of the routing in delay tolerant networks. Chapter 3 illustrates the use of game theory to study reputation systems and price systems, and analyze their underlying incentives and deficiencies. We also propose an integrated system that can achieve high cooperation incentives. Based on the theoretical results in Chapter 3, we further propose a distributed reputation management system with high scalability and low overhead in Chapter 4. Based on the system proposed in Chapter 4 for trustworthy routing, in Chapter 5, we study an efficient routing protocol/mechanism to increase the efficiency of the distributed wireless systems. In Chapter 6, we present the validation of the integrated model. We also show the simulation results of the ARM reputation management mechanisms and the selected case studies for efficient routing in DTN. Finally, we give our concluding comments and future work in Chapter 7.

Chapter 2

Related Work

How to detect and punish selfish nodes in distributed systems has been the subject of many research projects. Reputation systems and price systems are two main approaches proposed to encourage cooperation between mobile nodes in distributed wireless systems such as MANETs. Game theory is also used by many research communities as a mathematical tool to study human behaviors in wireless networks. How to efficiently route a message in delay tolerant networks also receives significant attention in the mobile network research community. In this chapter, we will introduce the related research of selfish node detection and punishment based on reputation and price systems. We will also present different methods that use game theory as mathematical tools for wireless network research. Finally, we will show different routing mechanisms that have been proposed for efficient message routing in delay tolerant networks. We will indicate the differences between the current state of the art and this dissertation research.

2.1 Reputation and Price Systems

Reputation Systems. A reputation system gathers observations of node behaviors and calculates node reputation values [14, 49, 50, 74, 76, 82, 69, 78, 83, 107, 67, 16]. The system detects and punishes low-reputed nodes by isolating them from the MANET. There are two types of reputation systems: first-hand based and second-hand based.

In first-hand based reputation systems [14, 49, 50, 74], a node only believes its own observations about the behavior of other nodes, and the exchange of reputation information between nodes is disallowed. OCEAN [14] is the first to argue that the second-hand reputation information is subject to false accusations and requires maintaining trust relationships with other nodes. Therefore, OCEAN attempted to see how far the system can perform by using direct first-hand observations of other node behavior. Conti *et al.* [49] proposed acknowledgment-based schemes, termed TWOACK and S-TWOACK, which are used for the source routing protocol. The destination node sends an acknowledge message for every received packet to the source node. In this way, the selfish nodes can be identified if the acknowledge messages from the destination nodes are not received by the source nodes. The source nodes maintain a blacklist of their observed selfish nodes and try to avoid the selfish nodes in the routing path in future routes. Liu *et al.* [50] proposed to further improve TWOACK and S-TWOACK by only sending acknowledge packets for a fraction of received data packets instead of all data packets to reduce the overhead. Dewan *et al.* [74] proposed a system in which every node maintains a reputation value of other observed nodes. Since every node maintains different routing paths to other nodes, the nodes choose the next hop nodes with a sufficiently high reputation during the packet routing to achieve routing reliability.

In second-hand reputation systems [76, 82, 69, 78, 83, 107, 67, 16], nodes share observations of node behaviors by periodically exchanging observed information. In Core [76], the

authors proposed to determine the final reputation of a node by combining the subjective reputation and second-hand reputation. Subjective reputation is the reputation calculated directly from a subject's observation. Indirect reputation is the reputation learned from the neighbors. CONFIDANT [82] aims to detect and isolate misbehaving nodes, thus making it unattractive to deny cooperation. Trust relationships and routing decisions are based on experienced, observed, or reported routing and forwarding behavior of other nodes. Marti *et al.* [69] introduced two functions *watchdog* and *pathrater* to mitigate the efforts of routing misbehavior. When a node forwards a packet, the nodes' watchdog verifies that the next node in the path also forwards the packet. If the next node does not forward the packet, then the next node is regarded as a misbehaving node. The misbehaving information is propagated throughout the network. The pathrater uses this knowledge of misbehaving nodes to choose the network path that is most likely to deliver packets. He *et al.* [78] proposed to objectively calculate the reputation of a node based on the packet forwarding ratio of the node associated with its confidence and credibility value. The propagation of reputation is efficiently secured by a one-way-hash-chain-based authentication scheme. Buchegger *et al.* [83] proposed to use a Bayesian approach for the representation and building of reputation as well as for subsequent decision-making depending on the reputation. They found that by excluding opinions that deviate substantially from first-hand observation and the majority opinion of second-hand opinions gathered overtime, the robustness of the reputation system remains intact even with a large number of liars in the network. Zong *et al.* [107] proposed a trust computation model based on artificial neural networks (ANN), which can help trust model tune its parameters automatically to adapt to various requirements on the trust calculation. They also proposed a broker-assisting information collection strategy based on the clustering method. With the support of brokers, the clustered sub-communities are managed by a reputation mechanism in an efficient and scalable manner to help the community members collect reputation values of others with high quality. Refaei *et al.* [67] proposed an adaptive

reputation management system that uses a time-slotted approach to allow the reputation evaluation function to quickly and accurately capture changes in node behavior for reputation calculation. A Sequential Probability Ratio Test (SPRT) is also used in reputation calculation to distinguish between cooperative and misbehaving neighbors. Buchegger *et al.* [16] proposed a reputation system that can cope with false disseminated information. In the system, every node maintains a reputation rating and a trust rating about everyone else that the node cares about and periodically exchanges the first-hand reputation information with others. They also proposed a modified Bayesian approach to only accept the second-hand reputation information that is not incompatible with the current reputation rating to avoid false ratings. Therefore, the trust ratings are updated based on the compatibility of second-hand reputation information with prior reputation ratings.

Although these proposed reputation systems try to use either linear reputation adjustment mechanisms [14, 49, 50, 74, 76, 82, 69, 78, 83] or non-linear reputation adjustment mechanisms [107, 67, 16] for reputation calculation, they still use a threshold to distinguish selfish nodes from cooperative nodes. Thus, clever selfish nodes can wisely maintain their reputation value just above the threshold by selectively forwarding others' packets regardless of the reputation calculation mechanism. Such nodes can take advantage of other cooperative nodes without being detected. Also, these methods cannot reward high-reputed nodes differently or punish low-reputed nodes in different reputation levels.

Price Systems. In price systems, nodes are paid for offering packet forwarding service and pay for receiving forwarding service. The payments can be in money, stamps, points or similar objects of value [56, 57, 66, 43, 54, 85, 37]. Buttyan *et al.* [56] perhaps is the first work to introduce a virtual currency in MANETs to stimulate a cooperative behavior and prevent congestion. If a mobile node wants to use a service, then it must pay for it in virtual credits. Therefore, the mobile nodes are no longer interested in sending useless messages to overload the network. They are motivated to provide services to other mobile nodes since this is

the only way to earn virtual credits. Based on the study in [56], Buttyan *et al.* [57] further proposed a security module maintaining a counter, called virtual credit counter. The counter is decreased when the node wants to send a packet as originator and increased when the node forwards a packet. The value of the counter must remain positive if a node wants to send a packet as originator. Jakobsson *et al.* [66] developed a micro-payments model for fostering collaboration among selfish (rational) participants in multi-hop cellular networks. Instead of mobile nodes, base stations are responsible for verifying that all packets are accompanied by a valid payment. An auditing process is further used to detect the cheating behavior of the mobile nodes. Crocraft *et al.* [43] considered the issue of how prices can be determined automatically by the ability of nodes to pay the costs for transmitting traffic. They also illustrated how network resources are allocated to users according to their geographical positions in the packet routing. Anderegg *et al.* [54] proposed Ad Hoc-VCG, in which every node bids to be relay nodes for packet transmission and a source node chooses the second best bid, which forces selfish nodes to reveal their true forwarding costs. The source node pays the relay node a premium over its actual cost after the packet is forwarded to encourage selfish nodes to truly offer forwarding services. A packet is always routed on a path with lowest accumulated costs. Zhong [85] proposed a cheat-proof, credit-based system for stimulating cooperation among selfish nodes in MANETs without any tamper-proof hardware at any node. When a node receives a message, the node keeps a receipt of the message and reports the receipt to the Credit Clearance Service (CCS) if a fast connection to a CCS is available. Based on the receipts, CCS can determine the charge and reward to each node involved in the transmission of a message. Janzadeh *et al.* [37] proposed a credit-based cooperation mechanism that does not rely on tamper-proof hardware. In the system, every source node uses a digital signature only in its first transaction with any cooperative intermediate node that forwards packet(s) for it. Any following transaction with such a node requires only hash operations instead of digital signature operations both for the source node and the

intermediate nodes, reducing the computation load of the mobile nodes. They use credits to provide appropriate incentives and fines to discourage rational nodes from misbehaving.

Although all of the proposed price systems can encourage nodes to be cooperative, most systems fail to provide a mechanism to measure measure/monitor the service quality of a node. Moreover, they fail to punish a selfish and wealthy node that earns many credits by being cooperative early but dropping packets later. Also, the nodes that need no forwarding services can always refuse to help others in forwarding packets.

Previous research did not analyze the effectiveness of cooperation incentives in the individual reputation and price systems. In this dissertation research, we analyze the cooperation incentive strategies in the individual reputation and price systems using game theory model. Based on the analysis results, we propose an integrated system and a hierarchical Account-aided Reputation Management system (ARM), which coordinately leverage the advantages of reputation systems and price systems to effectively defer selfish behaviors. In addition, relying on the collected global information by the DHT, ARM effectively detects false information and accurately calculates node reputation. With the efficient operation provided by DHT, ARM can also reduce periodical reputation information exchanges, the storage and computing burden of each node in the previous reputation systems as well as eliminates the virtual credit circulation in the network in the previous price systems.

2.2 Game Theory-based Works

Game theory has been used to optimize packet routing in MANETs and the allocation of resources including channel and power resources in wireless networks. These studies use either non-cooperative game theory [19, 99, 80, 45, 10, 51, 92, 64, 91] or cooperative game theory [100, 58, 6, 95, 21].

Non-cooperative Game Theory. Non-cooperative game theory has been used broadly

in the research of wireless networks in terms of distributed channel, spectrum and power resource allocation [19, 99, 80]. Saraydar *et al.* [19] presented a power control solution for wireless data in the analytical setting of a game theoretic framework. In this context, the quality of service (QoS) of a wireless node receives is regarded as the utility. The distributed power control is modeled as a non-cooperative game where users try to maximize their utility. Han *et al.* [99] attempted to motivate individual users to adopt a social behavior and enhance the system performance by sharing the resources. They considered both power control and adaptive modulation by designing non-cooperative games at both the user level and the system level. A non-cooperative power control game is designed at the user level to maximize user's transmitted power constraint. At the system level, the optimization goal is to maximize the overall system throughput under the maximal transmitted power constraint. Etkin *et al.* [80] studied a spectrum sharing problem in an unlicensed band where multiple networks, such as 802.11, bluetooth, and walkie-talkies, coexist and interfere with each other. They provided a unified framework to study the issues of efficiency, fairness and incentive compatibility in a non-cooperative spectrum sharing situation based on a non-cooperative game.

Numerous studies have also been proposed to use non-cooperative game theory to optimize the routing in MANETs [45, 10, 51, 92, 64, 91]. Jaramillo *et al.* [45] proposed a distributed and adaptive reputation mechanism for MANETs, which avoids a retaliation situation after a node has been falsely perceived as selfish so cooperation can be restored quickly. They also proposed a network model based on non-cooperative games to understand the impact of imperfect measurements on the robustness of some previously proposed reputation strategies. In the analysis, the schemes punish selfish behavior at the expense of decreasing the throughput of cooperative users, which may occasionally lead to complete network disconnection. Altman *et al.* [10] proposed a framework of non-cooperative game theory to provide incentives for collaboration in MANETs. The incentives provided in the

proposed framework are based on a less aggressive Tit-for-Tat punishment mechanism that can be implemented in a completely distributed manner. They assumed that if the fraction q' of packets forwarded by a mobile node is less than the fraction q forwarded by other mobile nodes, then this will result in a decrease of the forwarding probability of the other mobile nodes to the value q'. Kameda *et al* [51] examined the issue that the Nash Equilibrium (NE) action set is not the most profitable action set in a non-cooperative game for network traffic flow control. They found that in communication networks where each user decides its throughput by itself to optimize its own utility, the system throughput is not maximized. However, they did not provide a Pareto optimal strategy for optimal system performance. Srinivasan *et al.* [92] built a non-cooperative game to determine the Pareto optimal throughput in the system and the optimal throughput each node can achieve. To achieve the optimal throughput, they proposed a Pareto optimal strategy called Generous TIT-For-TAT (GTFT), which is used by the nodes to decide whether to accept or reject a relay request to optimize individual throughput. Felegyhazi *et al.* [64] proposed a model based on the game theory and graph theory to investigate the equilibrium conditions of packet forwarding strategies. They studied a number of interaction strategies and showed that it is very important to provide incentives for node cooperation. Urpi *et al.* [91] proposed an approach to choose interaction strategies based on Bayesian games, where the players are the nodes in the network. In the Bayesian games, prior to choosing its next action, a node has an opportunity to analyze the past behaviors of its neighbors and its consideration priorities on energy consumption and throughput to decide how to react in the next step. Although the paper presents a formal model to guide the selection of an interaction strategy that can optimize each individual node's benefit, it does not consider how much cooperation incentive the strategy can provide.

Cooperative Game Theory. Cooperative game theory is generally applied for the analysis of networks and spectrum sharing [100, 58, 44, 6, 95, 89]. Han *et al.* [100] applied cooper-

ative game theory for resource allocation in orthogonal frequency-division multiple access systems (OFDMA) to maximize the overall system rate, under the constraints of each user's minimal rate requirement and maximal transmitted power. First, they developed a two-user bargaining algorithm to negotiate the usage of subcarriers and then they grouped the users into groups of size two as a coalition. Within each coalition, they used a two-user algorithm to improve the system rate. Cao *et al.* [58] proposed a local bargaining approach where users are self-organized into bargaining groups and adapt their spectrum assignment to approximate an optimal spectrum assignment. They also proposed a fairness bargaining with feed poverty to improve fairness in a spectrum assignment and derived a theoretical lower bound on the minimum assignment each user can get from bargaining. Such a bound can be utilized to guide the bargaining process of spectrum assignment. Aram *et al.* [6] considered a network in which several service providers offer wireless access service to their respective customers. They modeled the cooperation using transferable payoff coalitional game theory to find the optimum resource allocation strategies. Saad *et al.* [95] studied cognitive radio networks where secondary users occupy the channels of primary users in an attempt to reduce the interference of secondary users on the primary users through collaborations. The authors modeled the problem as a cooperative game and proposed a distributed algorithm for coalition formation through simple merge and split rules.

Based on the previous research using game theory, in this dissertation research, we apply the cooperative and non-cooperative game theory to model the node interaction in reputation and price systems for cooperation encouragement and seeking strategies that can thwart the intentions of selfish and malicious nodes.

2.3 Routing Protocols

One method to deal with message routing in DTNs is to reinforce connectivity on demand by assigning a number of specialized nodes (e.g., robots and satellites) to fill the "communication gap" when a disconnection occurs [106, 79]. Zhao *et al.* [106] proposed a Message Ferrying (MF) approach for data delivery in sparse networks. MF is a proactive mobility-assisted approach, which utilizes a set of special mobile nodes called message ferries to provide communication services for nodes in the network. Message ferries move around the deployment area and take responsibility for carrying data between nodes. Li *et al.* [79] attempted to explore the possibility of changing the host trajectories to facilitate communication. Given an ad hoc network of mobile computers where the trajectory of each node is approximately known, they developed an algorithm for computing a trajectory for sending a message from host A to host B by recruiting intermediate hosts to help. However, these approaches, which require approximate knowledge of the network, are not applicable in a highly dynamic self-organized network. In our dissertation research, we have proposed a routing mechanism called SEDUM that works well in a highly dynamic self-organized network.

Epidemic routing (i.e., flooding) [94] is a widely-used routing strategy in DTNs. In this method, a message is flooded from a node to all its neighbor nodes in the system recursively to transmit a message to a destination node. This method requires that each node has a large buffer for storing messages in transmission. It can achieve a short delay by locating a shortest routing path at the cost of high network resource consumption. Some improved approaches are proposed to reduce the overhead of epidemic routing [86, 97, 105, 46, 96]. In [86], nodes remove redundant replicas of a message when the message has been transmitted by exchanging the "metadata" of delivered messages. The work in [105] uses a gossip algorithm, in which a message is forwarded to partial neighbors for the message forwarding. Additional studies [97, 46, 96, 23] further improved the mechanism proposed in

[105] by using network coding mechanisms to increase the routing reliability and reduce the routing overhead. Wang *et al.* [97] proposed to encode a message with erasure coding and spread them over multiple relays while using a fixed amount of overhead, which is much more robust to failures of a few relays or some bad choices compared to the mechanisms in [105]. In contrast to simply forwarding the information contained in the packets, Widmer *et al.* [46] proposed to let the nodes send out packets with linear combinations of previously received information based on the network coding methods. By compressing the information based on the network coding, the amount of messages exchanged in the system is greatly reduced, leading to much smaller system overhead. Wang *et al.* [96] proposed to use erasure coding technology to achieve a desired data delivery ratio with minimum overhead. Nodes determine whether to transmit or drop messages based on the importance of the messages. Chen *et al.* [23] proposed a hybrid routing method that combines erasure coding based routing and a replication technique to reduce the system overhead. Although all of these methods can improve the performance of the epidemic routing to a certain extent, the coding mechanisms still lead to a high resource consumption.

The other widely studied routing proposal for DTNs is single-copy routing, including direct routing [87] and probabilistic routing [9, 18, 17, 31, 47, 71, 27, 25].

Direct routing lets the source or a moving relay node carry a message all the way to the destination. Spyropoulos *et al.* [87] looked into a number of "single-copy" direct routing schemes such as direct transmission, "oracle-based" optimal algorithm, and randomized routing. They found that all these methods can significantly reduce the resource requirements of flooding-based algorithms and maximize the transmission capacity of the system if they are carefully designed. Although this method can maximize the transmission capacity of the system, it leads to long transmission delay.

Probabilistic routing uses different information to assist message routing. Lindgren *et al.* [9] proposed a probabilistic protocol for routing in intermittently connected networks

that use node encounters frequency and transitivity to enhance performance over previously existing protocols. Burns *et al.* [18] proposed a routing protocol that maintains a movement model of the network participants and uses this information to perform routing of messages on the network. It estimates the probability of a particular message being delivered by a given peer, and thus is capable of making informed routing decisions. They further used multi-objective control methods from robotics to generate motions capable of optimizing multiple network performance metrics simultaneously. Burgess *et al.* [17] proposed a probabilistic routing based on prioritizing both the schedule of packets transmitted to other peers and the schedule of packets to be dropped. These priorities are determined based on the delivery probability of the packets according to historical data as well as several complementary mechanisms, including acknowledgments, a head-start for new packets, and lists of previous intermediaries. These methods reduce the transmission overhead of epidemic routing at the cost of possible delivery delay due to suboptimal relay node choices. Henri *et al.* [31] pointed out that consulting the time period of a node since it encountered the destination node when making a forwarding decision results in superior performance over flooding. Jain *et al.* [47] proposed a forwarding algorithm to minimize the average delay of message delivery using oracles that know the entire topology of the current network. However, such oracles are very difficult to implement because of the high mobility and intermittent connections between nodes. The context-aware adaptive routing (CAR) protocol [71] periodically refines the prediction of node mobility in order to identify the cluster where the destination nodes belong to and select an optimal node as a message carrier to the destination nodes. It also uses a proactive routing algorithm such as Destination-Sequenced Distance Vector (DSDV) routing to publish the predicted delivery utility. Admittedly, CAR works well in a partially connected network. However, in a very sparse DTN, the update messages are unlikely to be published in a timely manner. Polo *et al.* [27] proposed a publish/subscribe system for DTNs. It relies on the Kalman filter [40] to predict the routing of nodes based on current

topology states. Conan *et al.* [25] presented a routing strategy based on single copy, in which a node relays a message to a neighbor that is closer in terms of total expected delivery time to the destination. The strategy uses the estimates of the average inter-contact times between the nodes in the network as the routing metric and limits the transmission hops to two. Although the single-copy routing strategy saves node resources and produces lower transmission overhead, it is likely to suffer from severe transmission delays if a suboptimal forwarding node (i.e., a node not in the shortest path between the source and the destination) is chosen.

Recently, a few routing protocols have been proposed that explore communities in social networks in MANETs and DTNs. LABEL [41] exploits clustering algorithms to group nodes into communities according to their affiliation through labels. They found that simply identifying community can improve message delivery. Li *et al.* [60] proposed to construct communities based on the neighboring relationships from node encounter histories in a distributed manner. They also proposed a locally weighted publish/subscribe method for data collecting, storage, and propagation within and among the communities. Ghosh *et al.* [35] proposed a sociological orbit aware location approximation and routing algorithm by extracting the mobility information of the users based on the observation that the movement of a mobile user exhibits a partially repetitive "orbital" pattern involving a set of "hub" nodes. Taking advantage of these hub nodes, the messages can be efficiently routed to the destination node compared to the flooding algorithms. Costa *et al.* [28] proposed a routing framework for publish-subscribe that exploits predictions based on metrics of social interaction (e.g., patterns of movements among communities) to identify the best information carriers. Daly *et al.* [30] proposed to identify some bridge nodes based on their centrality characteristics. They explore the concept of ego networks [15] for bridge node extraction where nodes are not required to exchange information about the entire network topology but only the locally available network information. Hui *et al.* [42] evaluated the impact of

community and centrality on packet forwarding, and proposed a hybrid algorithm, BUBBLE, that selects high centrality nodes and community members of destination as relays. Gao *et al.* [34] studied multicast in DTNs from the social network perspective and investigated the essential difference between multicast and unicast in DTNs. They also formulated relay selections for multicast as a unified knapsack problem by exploiting node centrality and social community structures in the networks. Chen *et al.* [24] proposed a hybrid routing scheme combining utility and centrality metrics to make forwarding decision. The utility and centrality metrics are defined with two kinds of contact history, the ages of last encounter and the cumulative contact durations between the nodes in the network, respectively.

Based on the previous research on routing, in this dissertation research, we propose a Social nEtwork and utility based DistribUted Multi-copy routing protocol (SEDUM). SEDUM is different from the previous research in two aspects. First, rather than relying on either flooding or single-copy routing, it uses multi-copy for routing based on the optimal tree replication algorithm to achieve a tradeoff between routing delay and overhead. Second, SEDUM uses a novel duration utility that can automatically capture the social movement features without social graph analysis and community detection/construction as in the previous research. Although Li and Chen [60, 24] also used contact durations in routing utilities calculation, they did not show the rationale of using contact duration utility and the advantage of contact duration utility over contact frequency utility. Meanwhile, their method is not suitable for a system with dynamically changing network size. Even the network size and network patterns change dramatically, SEDUM still can achieve a high performance with its optimal tree replication mechanism and buffer management algorithm.

2.4 Summary

In this chapter, we have discussed the research regarding reputation and price systems for selfish node detection and punishment, game theory modeling, and routing in DTNs. We have also explained the difference and relevance between our work and the previous research. In the next chapter, we will introduce our proposed method to increase the trustworthiness of the distributed wireless systems.

Chapter 3

Trustworthiness: Analysis of Cooperation Incentive Strategies

In this chapter, we propose mechanisms to solve the first challenge in distributed wireless systems. That is, cooperation incentives that effectively encourage nodes to offer services and thwart the intentions of selfish and malicious nodes. We describe theoretical studies of the trustworthiness of the reputation and price systems, and analyze their underlying incentives and deficiencies for cooperation encouragement. We also present our proposed integrated system that can provide high incentives to encourage cooperation. We assume that all nodes in the system are self-interested. That is, they always try to choose actions that maximize their own benefit. We intend to answer the following questions:

(1) Is it possible to encourage the nodes in a system to be cooperative without any cooperation incentive strategy? (Section 3.2)

(2) How effective are the cooperation incentives provided by existing individual reputation (Section 3.3) and price systems (Section 3.4)? What are the deficiencies of individual reputation (Section 3.3) and price systems (Section 3.4)?

(3) How to design a system that can overcome the deficiencies of individual reputation and price systems, and provide higher cooperation incentives? (Section 3.5)

To answer these questions, we build different game theory models including cooperative and non-cooperative game models to study the cooperation incentives provided by (i) a system without any cooperation incentive strategy (defenseless system), (ii) a reputation system and (iii) a price system. We use both cooperative game and non-cooperative game to investigate the best strategy for each node to maximize its benefit.

Based on the cooperative game model, we find that in all these systems, each node earns the maximum benefit only when they form a grand coalition, in which all nodes in the system are cooperative. However, in order to form such a coalition, the cooperative strategy should be enforced by a third party that can monitor or control the users. However, such an assumption cannot always be valid especially for the commercial applications in our daily life, which are controlled by several authorities. Based on the non-cooperative game model, we find that the cooperation incentives provided by both reputation systems and price systems are limited. The strategies of using a threshold to determine the trustworthiness of a node in the reputation system and the strategies of rewarding cooperative nodes in the price system may be manipulated by clever but selfish nodes. Specifically, the reputation systems treat nodes with reputation values higher than the threshold equally. Thus, a node can keep its reputation value just above the threshold to receive the same benefit as nodes with much higher reputations. This behavior is unaffected by the reputation calculation mechanisms and can exist in all reputation systems with the threshold strategy. The price system lacks an effective method to detect a selfish and wealthy node that earns many credits by cooperating initially but becomes non-cooperative (selfish and non-cooperative are interchangeable in this dissertation) later without receiving a penalty.

Inspired by our observations, we propose an integrated system to leverage the advantages of both reputation and price systems by integrating the misbehavior detection

mechanism from the reputation system and the cooperation incentive mechanism from the price system. The integrated system can also overcome their individual disadvantages. The theoretical analysis shows that the integrated system can provide higher cooperation incentives than the individual reputation and the price systems in terms of higher payoffs for the cooperation strategy and make the cooperative strategy to be the Nash Equilibrium and Pareto-optimal. The system is also more effective in selfish node detection, which can greatly reduce the packet dropping rate of the whole system.

The remainder of this chapter is organized as follows. Section 3.1 introduces the basic game theory models. Section 3.2, Section 3.3, Section 3.4 present the game theory based analysis for the individual defenseless, reputation, and price systems. Section 3.5 describes and analyzes the proposed integrated system with the game theory model. Section 3.6 summarizes this chapter. The experimental results are given in Chapter 6.

3.1 Game Theory Models For Mobile Ad hoc Networks

3.1.1 Classification of Basic Game Theory Models

Game theory is an area of applied mathematics that models and analyzes a system in which every individual attempts to find the best strategy for success depending on the choices of others in node interactions. As shown in Figure 3.1, game theory models can be generally categorized as *cooperative games* or *non-cooperative games.* In a cooperative game, the nodes agree on the strategy and this strategy cannot be altered. In contrast, nodes in non-cooperative games can change their strategies at any time to maximize their benefits. Non-cooperative games can be further classified into one-interaction games and repeated games. In the former, individuals only interact with each other once. In the latter, individuals interact with each other multiple times.

Repeated games can be further classified into finite repeated games or infinite repeated

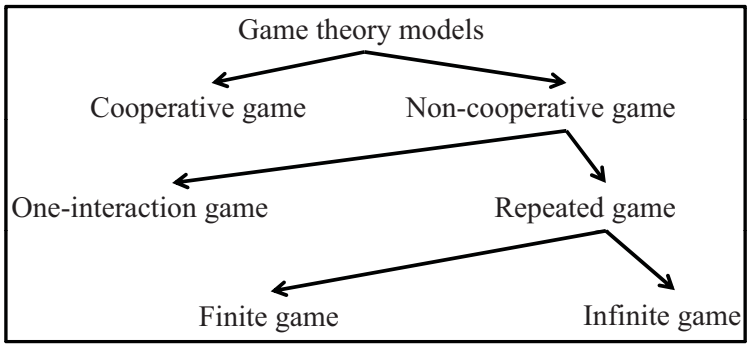


Figure 3.1: Classification of game theory models.

games. In finite repeated games, there are a finite number of interactions for a pair of players, while in infinite repeated games there are no restrictions on the number of interactions. Game theory provides analytical tools to predict the outcome of complex interactions among rational and self-interested entities who always attempt to reach the best outcome [93]. Table 3.1 shows the parameters used in the analysis.

3.1.2 Non-cooperative Game in MANETs

Regarding nodes in MANETs as rational and self-interested entities, a game theory model can be built. We use $\mathcal{N} = \{1, 2, ..., n\}$ to denote the set of all mobile nodes (i.e., game players) in a routing path. In an interaction between a pair of nodes in routing, each node requests the other node forward a packet, and the other node either forwards the packet or drops the packet. We use A_i to denote the action set for node i, and $A_i = \{I, C\}$; the C (i.e., cooperative) action means the node is willing to help the other node forward a packet, while the I (i.e., incooperative, non-cooperative) action means it drops the packet. Action and strategy are interchangeable terms in this dissertation. The action chosen by node i is denoted by a_i, and the actions chosen by other nodes are denoted by an action set $\mathbf{a_{-i}} = \{a_1, a_2, a_3, ...a_{i-1}, null, a_{i+1}, ..., a_n\}$. $\mathbf{a} = (a_i, \mathbf{a_{-i}}) = \{a_1, a_2, a_3, ...a_{i-1}, a_i, a_{i+1}, ..., a_n\}$ denotes the action set of all nodes on a path for the routing of one packet. If any node is

Table 3.1: Parameters used in the analysis.

c	packet forwarding cost	m_r	packet forwarding reward
m_p	packet forwarding price	n_d	the number of dropped packets
n_g	the number of generated packets	n_r	the number of received packets
p	packet forwarding benefit	v	characteristic function
x	allocated payoff	A	action set
$\mathcal{N}$	the set of total mobile nodes	$\mathcal{S}$	a subset of mobile nodes
U	payoff	P_d	average packet drop probability
P_s	the probability of an account state	R	current reputation value
T_R	reputation threshold	V	current account value
I	non-cooperative action	C	cooperative action

uncooperative, the packet will be dropped. We use D to denote the Cartesian product of the action set for a node, use $U_i(a_i, \mathbf{a_{-i}})$ to denote the utility (i.e., payoff, benefit) function of a node i given the strategies used by other nodes, and use $U(\mathbf{a})$ to denote the sum of the utilities of all nodes. The game theory model for MANETs given a normal form of game G is,

$$G =< \mathcal{N}, D, U_i(a_i, \mathbf{a_{-i}}) > \tag{3.1}$$

Every rational node in the system intends to choose an action that maximizes its utility for a given action tuple of the other nodes. That is, the best action $a_i^* \in A_i$ is the *best response* of node i to $\mathbf{a_{-i}}$ iff for all other $a_i \in A_i$, $U_i(a_i^*, \mathbf{a_{-i}}) \geq U_i(a_i, \mathbf{a_{-i}})$.

Definition 1. A **Nash Equilibrium (NE)** is an action tuple that corresponds to the mutual best response. Formally, the action tuple $\mathbf{a}^* = (a_1^*, a_2^*, a_3^*, ..., a_n^*)$ is a NE if $U_i(a_i^*, \mathbf{a^*_{-i}}) \geq U_i(a_i, \mathbf{a^*_{-i}})$ for $\forall a_i \in A_i$ and $\forall i \in \mathcal{N}$ [93], where A_i denotes the action set (cooperative, non-cooperative) for node i. Therefore, a NE is an action set where no individual rational node can benefit from unilateral deviation.

Definition 2. An outcome of a game is **non-Pareto-optimal** if there is another outcome that would give both players higher payoffs, or would give one player the same

payoff but the other player a higher payoff. An outcome is **Pareto-optimal** if there are no other such outcomes [73].

Based on Definitions 1 and 2, we know that encouraging the cooperation between nodes is essentially enforcing the cooperation strategy between nodes to become NE and Pareto-optimal.

Specifically, for the nodes in the routing path from source node 1 to destination node n, since the packet forwarding interaction only occurs between two neighboring nodes,

$$\mathbf{a} = (a_1, a_2, ..., a_n) = \cup_{i=1}^{n}(a_{i-1}, a_i). \tag{3.2}$$

To ensure that $(C_1, C_2, ...C_n)$ is NE, according to Definition 1, we must guarantee that $\forall\ i$ $(i \in [1, n])$, $U(C_1, ..., C_i, ..., C_n) \geq U(C_1, ..., a_i, ..., C_n)$. For the interaction between two neighboring nodes $i-1$ and i, if the cooperation strategy (C_{i-1}, C_i) is not NE, then $U(C_{i-1}, C_i) \leq U(a_{i-1}, C_i)$. Based on Equation (3.2), $U(C_1, ..., C_i, ..., C_n) \leq U(C_1, ..., a_i, ..., C_n)$, which contradicts our assumptions. To ensure that $(C_1, C_2, ...C_n)$ is Pareto-optimal, according to Definition 2, we must guarantee that $U(C_1, ..., C_i, ...C_n) \geq U(a_1, ..., a_i, a_n)$. For the interaction between two neighboring nodes $i-1$ and i, if the cooperation strategy (C_{i-1}, C_i) is not Pareto-optimal, $U(C_{i-1}, C_i) \leq U(a_{i-1}, a_i)$. Based on Equation (3.2), we can obtain $U(C_1, ..., C_i, ...C_n) \leq U(a_1, ..., a_i, a_n)$, which also contradicts our assumptions. Therefore, to ensure that $(C_1, C_2, ...C_n)$ is NE and Pareto-optimal, we must ensure that the interaction strategy between two neighboring nodes in the routing path is NE and Pareto-optimal.

In this dissertation, we consider the cooperation of nodes along one routing path to forward one packet. For the case in which multiple nodes transmit multiple packets to the same next hop node, we can separately consider the interactions between the multiple nodes and the next hop node for different packets. Prior to describing the models, we first use an example to explain the non-cooperative game.

Non-cooperative Game Example. Table 3.2 shows an example of a payoff matrix for a non-cooperative game with a two-node interaction. If node i is cooperative, node j's payoff is 4 when it is cooperative and 6 when it is not cooperative. Hence, node j chooses the I strategy. If node i is non-cooperative, node j's payoff is 0 when being cooperative and 1 when being non-cooperative. Thus, node j still chooses the I strategy. As a result, no matter which strategy node i selects, being non-cooperative produces more utility than being cooperative for node j, i.e., $U_j(I, \mathbf{a_{-j}}) > U_j(C, \mathbf{a_{-j}})$. Similarly, no matter which strategy node j chooses, being non-cooperative generates more utility than being cooperative for node i, i.e., $U_i(I, \mathbf{a_{-i}}) > U_i(C, \mathbf{a_{-i}})$. We use I_i and C_i to represent the cases that node i takes for the I and C actions, respectively. In this game, the action set (I_i, I_j) dominates other action sets. We say that (I_i, I_j), marked with a star in the table, is the NE of this game. From the payoff matrix, we can see that no individual node can obtain more benefit by unilaterally deviating from the action set (I_i, I_j). However, the payoff of the action set (I_i, I_j) is not the best outcome of the payoff matrix; the optimal payoff $(4, 4)$ is brought by the action set (C_i, C_j). (C_i, C_j) is the Pareto-optimal of this game. An effective cooperation incentive system should aim to achieve both the NE and Pareto-optimal outcomes rather than only the NE outcome.

Table 3.2: An example for non-cooperative games.

	Node j		
Node i		Cooperative	Non-cooperative
	Cooperative	(4,4)	(0, 6)
	Non-cooperative	(6, 0)	$(1, 1)^*$
Note: * denotes the Nash Equilibrium strategy set of this game			

3.1.3 Cooperative Game in MANETs

Definition 3. In cooperative games, the *characteristic function* describes how much

collective payoff a set of players gain by forming a coalition. The collective Pareto-optimal payoff is denoted by $v(\mathcal{S})$, where $\mathcal{S} \subseteq \mathcal{N}$ is a subset of total players. $v(i)$ is the characteristic function of player i in no coalition with other nodes (i.e., single member coalition) [32].

The single member coalition in the cooperative game is equivalent to the non-cooperative strategy in the non-cooperative game. $v(i)$ equals the NE payoff of player i in the uncooperative game.

Definition 4. Let x_i be the payoff received by player i ($i \in \mathcal{S}$). A vector $\overrightarrow{x} = (x_1, ...x_n)$ is a rational utility allocation if (1) $x_i \geq v(i)$ and (2) $\sum_{i=1}^{n} x_i = v(\mathcal{N})$ [32]. Definition 4 implies that a rational utility allocation should guarantee that a node earns more payoffs by forming a coalition with other nodes (Condition (1)). Also, the total allocated payoff of all players in a coalition should equal the collective Pareto-optimal payoff of all players (Condition (2)). Therefore, a node prefers to join a coalition that will bring a greater payoff than the single member coalition. Also, a node prefers to choose an optimal coalition from a number of coalition options. In non-rational utility allocation, a node may either choose not to join a coalition or to leave its current coalition to gain higher payoff from another coalition.

Definition 5. A coalition is deemed stable when no other coalitions can yield a higher payoff for each individual player in the stable coalition.

Cooperative Game Example. We use the same example in Table 3.2 to explain the cooperative game by analyzing the interactions between the players in the game. We show whether nodes sometimes have an incentive to form a coalition to optimize their utilities, how the nodes form a coalition, and whether the utility allocation to each node in the coalition is reasonable. According to the payoffs shown in Table 3.2, in a single member coalition $v(i)$=1 and $v(j)$=1. However, if player i and player j decide to form a coalition and ask a third party to enforce their strategies (i.e., the (C_i, C_j) strategy set is formed), the maximum payoff of the coalition is $v(i,j) = 8 > v(i) + v(j) = 2$. Also, for the payoff allocation in the

coalition, $x_i > v(i)$ and $x_j > v(j)$. That is, forming a cooperative coalition can bring more benefits to the nodes than forming a single member coalition. The (C_i, C_j) coalition is stable since no other coalitions can bring more benefit.

In the following sections, we build the game theory models introduced in this section for a defenseless MANET, a MANET with a reputation system, a MANET with a price system, and a MANET with an integrated system. We rely on the models to analyze the effectiveness of the cooperation incentives in each of the systems.

3.2 Game Theory Model for Defenseless Systems

We assume that a packet is the basic transmission unit between two nodes. When node i interacts with node j, node i sends a packet to node j and node j sends a packet to node i. The packet receiver can then choose to forward or drop the packet. If it chooses to forward the packet, it consumes resources for receiving, processing, and transmitting the packet. The resource consumption cost of forwarding a packet depends on several factors including channel condition, file size, modulation scheme, and transmission inference. As a generic model, we use c to denote the resource consumption cost for a node to forward a packet, and use p to denote the benefit gained by a node after its packet is forwarded by another node. In practice, c and p differs for nodes under different conditions and configurations. We assume that c and p can be generalized to the same measurement units. The benefit p includes the units of benefit gained by a node when its packet is successfully forwarded and the units of resources used for forwarding the packet. Thus, we assume $p > c$, since it is not rational for a user to use a device with $p \leq c$. We use p and c to represent the utility values in the game theory models for the cooperation incentive analysis. Then, the payoff for each node when both nodes are cooperative in an interaction is $(p - c)$. If one node is non-cooperative in transmitting a packet and the other is cooperative in transmitting a

packet, the selfish node earns a profit of p while the cooperative node earns a profit of $-c$. The reason is that the selfish node's packet has been forwarded by the cooperative node, but the selfish node has not forwarded the cooperative node's packet. If both nodes are non-cooperative in forwarding packets, the payoff of this action set is $(0, 0)$ because both nodes gain no benefits and consume no resources.

3.2.1 Non-cooperative Game for Defenseless Systems

One-interaction Game. Based on the cost and benefit of forwarding a packet between a pair of nodes in an interaction, we build a one-interaction game model as shown in Table 3.3. The table shows the payoff matrix for each combination of different actions taken by node i and node j. From the table, we see that since $p > p - c$ and $-c < 0$, independent of the strategy node j chooses, I is the best strategy for node i. Since $p > c$, independent of the strategy node i takes, I is also the best strategy for node j. Therefore, the action set (I_i, I_j) is the NE in this interaction. However, (C_i, C_j) is the optimal outcome since it leads to a payoff of $(p-c, p-c)$, which is much higher than $(0, 0)$. In this payoff matrix, the NE is not the Pareto-optimal. The nodes do not choose the Pareto-optimal action set because every node in the system is independent and self-interested, and each node in a pair does not know which action the opponent will take. If one node chooses C but the other chooses I, the payoff for the cooperative node will be the lowest. Therefore, the self-interested nodes will normally choose the safest strategy over the strategy that may lead to the best outcome [73] in a one-interaction game, at risk of a higher cost.

Repeated Games. Since in a real system the interactions between nodes are repeated, we also analyze the cooperation incentives in repeated games. Different from a one-interaction game, a player in repeated games learns the action history of other nodes, which helps it to make subsequent choices.

TIT-For-TAT has been recognized as the most effective interaction strategy thus far

Table 3.3: Payoff matrix for defenseless systems.

	Node j		
Node i		Cooperative	Non-cooperative
	Cooperative	(p-c, p-c)	(-c, p)
	Non-cooperative	(p, -c)	$(0, 0)^*$
Note: * denotes the Nash Equilibrium strategy set of this game			

for repeated interaction games [73]. In TIT-For-TAT, given a pair of nodes i and j, node i is initially cooperative with node j. If node j is also cooperative, node i will continue to use strategy C. Whenever node j is non-cooperative, node i will immediately become non-cooperative. Since (C_i, C_j) is Pareto-optimal, node i will forgive node j's non-cooperative behavior and periodically check whether node j wants to be cooperative again. An iterative (i.e., repeated) defenseless system (IDS) with TIT-For-TAT can effectively encourage node cooperation in an infinite game. The fundamental reason is that repeated games can change the Pareto-optimal strategy in the payoff matrix to NE when nodes interact with each other for infinite time; based on the interaction history of the opponents, the players can adjust their action strategy to be the Pareto-optimal to maximize their benefits. For a pair of nodes i and j in an infinite game, even though node i may lose some benefit by being cooperative at first, when node j is uncooperative, its cooperation will stimulate node j to be cooperative later when node j find being cooperative is more beneficial. Node i will also gain a much higher payoff for itself. Thus, by punishment (being non-cooperative) and forgiveness (being cooperative), a node can earn a high payoff in the long term.

However, IDS with TIT-For-TAT cannot encourage node cooperation in a finite game when the number of interactions is unknown to both nodes. Essentially, (C_i, C_j) is Pareto-optimal but not NE in IDS, i.e., the strategy I always dominates the strategy C. In a finite game, the best strategy for a node is to continue being cooperative and deviate in the last round from (C_i, C_j) if it knows when the interaction will end. For a node that

wishes to use the best strategy but does not know when the opponent will leave, it may suspect that the opponent will leave in the next round. Thus, the trust relationship between the interacting nodes will deteriorate. The only resolution to this problem is to make the (C_i, C_j) action set both NE and Pareto-optimal. In this situation, each node gains the same payoff or even higher payoff when its opponent deviates from its current action. Thus, each node has no incentive to deviate from the current cooperation strategy and is not afraid of the other node's deviation at any time during the interaction. One feature of repeated games is that they can change the Pareto-optimal strategy in a payoff matrix to be NE when nodes interact with each other for infinitely many times. Since the nodes in a MANET may randomly leave or join the network, the interaction between two nodes is actually a finite game with unknown number of interactions. In this situation, TIT-For-TAT cannot provide incentives for node cooperation. Therefore, the only method to encourage node cooperation in a MANET is to make (C_i, C_j) both NE and Pareto-optimal. In this case, regardless of whether node j deviates from (C_i, C_j) or not, the payoff received by node i will not be reduced, rather it will always be increased by choosing the cooperate strategy. Therefore, in MANETs, providing incentives for node cooperation essentially involves making (C_i, C_j) be both NE and Pareto-optimal in the payoff matrix.

Also, IDS with TIT-For-TAT can only provide the best action strategy for a node to obtain the best benefit based on the action of other nodes, but cannot monitor, detect or punish the misbehaving nodes efficiently. If node j is always uncooperative, node i can be non-cooperative or sometimes be cooperative to j. Node j will not be punished.

3.2.2 Cooperative Game for Defenseless Systems

In military applications, the nodes work for the US government, which can serve as the third party to monitor the nodes. In commercial applications, a telecommunication company could serve as the third party for monitoring. However, as many individual telecommunica-

tion companies exist, it is difficult to form a rule to monitor the nodes. Suppose the players can enforce contracts on each other through a third party and form a coalition to maximize their individual utilities. Take a three-node based coalition as an example. The coalitions that node i can choose include $\{i\}, \{i,j\}, \{i,k\}$ and $\{i,j,k\}$. Since all nodes in the system are identical, they have the same strategy options as node i. If node i does not form any coalition with other nodes, the interaction with another node is just the non-cooperative game. From the two-node interaction matrix shown in Table 3.3, we know that all nodes will choose the non-cooperative strategy. Therefore, $v(i)$ equals the NE payoff of player i in the non-cooperative game, that is $v(i) = 0$ (Definition 3). Below, we analyze the $\{i,j\}$ coalition. When player i and player j form a coalition, they choose the Pareto-optimal strategy (C_i, C_j) to interact with each other. Thus, the collective payoff of the $\{i,j\}$ coalition from their interaction is $(2p - 2c)$. Since player k chooses the NE strategy to interact with each of them, the collective payoff of the $\{i,j\}$ coalition from the interaction with k is $(-2c)$. Therefore, the collective payoff of the $\{i,j\}$ coalition is $v(i,j) = max\{2p - 4c, 0\}$. Similarly, $v(i,k) = max\{2p - 4c, 0\}$ and $v(i,j,k) = 6(p - c)$. Therefore, $v(i,j,k) = 6(p - c)$ is the highest utility the players can achieve when they form a grand coalition, in which all nodes in the system are cooperative. Since $x_i = x_j = x_k = 2(p - c) > v(i) = v(j) = v(k) = 0$ and $\sum_{i=1}^{n} x_i = v(\mathcal{N})$, according to Definition (3), the payoff allocation resulting from the grand coalition is rational.

Proposition 3.2.1 *In the cooperative game, the grand coalition with the* $(C_1, C_2, ...C_n)$ *action set is the only stable coalition.*

Proof Table 3.3 shows that the (C_i, C_j) action set leads to the Pareto-optimal payoff and (I_i, I_j) leads to the NE payoff. Therefore, in the n-node cooperative game, the action set $(C_1, C_2, ..., C_n)$ leads to a Pareto-optimal payoff. According to Definition 2, the Pareto-optimal action set $(C_1, C_2, ..., C_n)$ has the highest collective payoff. Since no other coalition

can generate a higher payoff, according to Definition 4, the $(C_1, C_2, ...C_n)$ action set is a stable coalition. Because $v(\mathcal{S}) < \sum_{i \in \mathcal{S}} x_{i_N}$ for all $\mathcal{S} \subset \mathcal{N}$, where x_{i_N} is x_i for node i in the grand coalition. Therefore, the grand coalition is the only stable coalition.

In conclusion, in a defenseless system, if the strategies of the nodes can be enforced by a third party, cooperative packet forwarding is the best choice for all rational nodes. However, such assumptions cannot always be valid especially for existing commercial applications that are controlled by several authorities.

3.3 Game Theory Model for Reputation Systems

One-interaction Game. Most reputation systems, regardless of whether the reputation value changes linearly or non-linearly, use a reputation threshold to distinguish selfish nodes from cooperative nodes. If nodes are cooperative in packet forwarding, the reputation values of these nodes are increased. If nodes are found to be uncooperative, their reputation values will be reduced. When the reputation value of a node is below threshold T_R, it will be detected as a selfish node.

Based on the packet forwarding benefit, cost, and reputation threshold, we build a one-interaction game theory model for reputation systems as shown in Table 3.4 along with Equations (3.3) and (3.4). From Table 3.4, we can see that when the reputation value of the node is above T_R, the non-cooperative action set (I_i, I_j) with payoff $(0, 0)$ is NE, but (C_i, C_j) is Pareto-optimal. When the R of a node is below T_R, all other action sets except the (C_i, C_j) action set produce $(0, 0)$ payoff. Therefore, only when the reputation value of the node is below T_R does the (C_i, C_j) become both NE and Pareto-optimal. In this situation, as no individual rational node can benefit from the unilateral deviation of (C_i, C_j), (C_i, C_j) becomes NE. As it can bring the maximum benefit for each node, (C_i, C_j) is also Pareto-optimal. Based on the above analysis, we can surmise the following proposition.

Table 3.4: Payoff matrix for reputation systems.

	Node j		
Node i		Cooperative	Non-cooperative
	Cooperative	(p-c, p-c)	$U(C_i, I_j)$
	Non-cooperative	$U(I_i, C_j)$	(0, 0)

$$U(C_i, I_j) = \begin{cases} (-c, p) & \text{if } R_{I(j)} > T_R \\ (0, 0) & \text{if } R_{I(j)} \leq T_R \end{cases} \tag{3.3}$$

$$U(I_i, C_j) = \begin{cases} (p, -c) & \text{if } R_{I(i)} > T_R \\ (0, 0) & \text{if } R_{I(i)} \leq T_R \end{cases} \tag{3.4}$$

Proposition 3.3.1 *Given a pair of nodes i and j in a reputation system, if their reputation values are larger than the reputation threshold T_R, the (I_i, I_j) strategy is NE and the (C_i, C_j) strategy is Pareto-optimal. If the reputation value of either node in the pair is less than T_R, the (C_i, C_j) strategy is both NE and Pareto-optimal.*

Proposition 3.3.2 *The cooperation incentive strategy provided by reputation systems will result in a situation where the node reputation values are near the reputation threshold.*

Proof As the payoff matrix in Table 3.4 shows, when the reputation value of a node is above T_R, (I_i, I_j) is the NE. Therefore, the node has incentive to be non-cooperative. Then, its reputation value continues to decrease as:

$$\lim_{R \to T_R^+} R = T_R. \tag{3.5}$$

When a node's reputation value is below T_R, (C_i, C_j) is the NE. Hence, the node will cooperate to increase its reputation value. The value continues to increase as:

$$\lim_{R \to T_R^-} R = T_R. \tag{3.6}$$

Consequently, the reputation values of nodes converge at T_R, meaning the nodes tend to keep their reputation value near the threshold value.

Proposition 3.3.2 implies that reputation systems cannot provide incentives to encourage nodes to be more cooperative when their reputations are close to and above T_R; it can only encourage nodes not to misbehave. Therefore, nodes are only motivated to keep their reputation values close and above the reputation threshold. If a node cleverly manipulates this policy by accepting partial transmission requests to keep its reputation just above the threshold, the performance of the system is impeded due to the packet drops.

We use R to denote the current reputation value of a node. We assume that in the first n_r packets that a node has received, it drops n_d packets, and forwards n_r-n_d packets. We use ΔR^+ to denote the reputation increase rate, the increased reputation value for a cooperation action, and use ΔR^- to denote the reputation decrease rate, the decreased reputation value for a non-cooperation action.

Proposition 3.3.3 *If a selfish node manages to keep its reputation value closely above the threshold, the upper bound of the packet drop rate P_d is:*

$$P_d \geq \frac{\Delta R^+}{\Delta R^+ + \Delta R^-}. \tag{3.7}$$

Proof Suppose that in the first n_r interactions, a selfish node can choose the I strategy for n_d interactions before its reputation value falls below T_R. Therefore,

$$n_d \cdot \Delta R^- - (n_r - n_d) \cdot \Delta R^+ \geq R - T_R, \tag{3.8}$$

$$\Rightarrow P_d = \frac{n_d}{n_r} \geq \frac{\frac{R-T_R}{n_r} + \Delta R^+}{\Delta R^+ + \Delta R^-}, \tag{3.9}$$

$$\Rightarrow \lim_{n_r \to \infty} P_d \geq \lim_{n_r \to \infty} \frac{\frac{R-T_R}{n_r} + \Delta R^+}{\Delta R^+ + \Delta R^-} = \frac{\Delta R^+}{\Delta R^+ + \Delta R^-}. \tag{3.10}$$

Proposition 3.3.3 implies two points. First, in a MANET with a reputation system, the packet drop rate of rational nodes is determined by the reputation increase rate for cooperative behavior and the decrease rate for non-cooperative behavior. Second, the packet drop rate is irrelevant to the threshold value. Therefore, to reduce the packet drop rate, a reputation system should have a low reputation increase rate and a high reputation decrease rate.

Propositions 3.3.2 and 3.3.3 show that the reputation system can only provide incentive to encourage nodes to keep their reputation values just above the reputation threshold, rather than encouraging them to be more cooperative in packet forwarding. Once a node has an R just above the threshold, it can always be served in the packet transmission. The reputation system treats all the nodes whose Rs are above the reputation threshold identically, regardless of their different cooperative levels. Therefore, a reputation system must have a complementary method to encourage all nodes to be highly cooperative with each other and differentially reward nodes in different altruistic levels.

Repeated Games. In a repeated game of reputation systems, for a pair of nodes i and j the Pareto-optimal action set alternates between (C_i, C_j) and (I_i, I_j) because the reputation values of the nodes fluctuate near T_R. Since (C_i, C_j) cannot always be the NE, the nodes will not always choose (C_i, C_j). Therefore, reputation systems cannot always encourage nodes to be cooperative in repeated games.

3.4 Game Theory Model for Price Systems

One-interaction Game. A price system uses virtual cash such as credits to encourage node cooperation in the system. If a node does not have enough credits for packet forward-

ing, all of its transmission requests will be rejected. In addition to the transmission cost c and transmission benefit p, we introduce credit payoffs m_r and m_p for service transactions. m_r denotes the packet forwarding reward in credits for one cooperative forwarding behavior and m_p denotes the packet forwarding price in credits for one forwarding service. In an interaction between a pair of nodes i and j with the strategy set (C_i, I_j), node j drops node i's packet and node i forwards node j's packet. Although the selfish node j can save the transmission cost c by refusing to forward node i's packet, it still should pay m_p for node i's forwarding service for its packet. On the other hand, although the cooperative node i loses packet transmission payoff p as its packet has been dropped by node j, it can still earn payoff m_r due to its cooperative behavior for forwarding node j's packet. Based on the packet forwarding benefit, cost, price and reward, we build the one-interaction payoff matrix for a pair of interaction nodes in a price system, as shown in Table 3.5, where $\triangle m = m_p - m_r$. In one interaction, both nodes cooperatively forward each other's packet. For the (C_i, C_j) strategy set, since both nodes are cooperative in the packet routing, they both earn the payoff p and spend c for the packet transmission. Also, since each node should pay m_p for the packet forwarding by the other and earn m_r for its own cooperative behavior, the payoff for (C_i, C_j) is (p-c-m_p-m_r, p-c-m_p-m_r). Similarly, the payoff for (C_i, I_j) and (I_i, C_j) can be calculated as shown in Equations (3.11) and (3.12). For example, in the (C_i, I_j) action set, since node i is cooperative to forward packets but its packet is not forwarded by node j, the payoff for node i is m_r-c. Meanwhile, since the packets of node j are forwarded by node i, node j should pay for the forwarding. Therefore, the payoff for node j is p-m_p. V_i and V_j denote the account value (i.e., credit amount) of node i and j respectively. When $V_i < 0$ or $V_j < 0$, there is no interaction between the nodes. Therefore, the payoff is (0, 0).

Proposition 3.4.1 *Price systems can make (C_i, C_j) NE iff the transmission cost c, transmission benefit p, packet forwarding price m_p, and packet forwarding reward m_r satisfy $p > m_p$ & $m_r > c$.*

Table 3.5: Payoff matrix for price systems.

	Node j		
Node i		Cooperative	Non-cooperative
	Cooperative	(p-c-$\triangle m$, p-c-$\triangle m$)	$U(C_i, I_j)$
	Non-cooperative	$U(I_i, C_j)$	$(0, 0)$

$$U(C_i, I_j) = \begin{cases} (-c + m_r, p - m_p) & \text{if } V_j > 0 \\ (0, 0) & \text{if } V_j < 0 \end{cases} \tag{3.11}$$

$$U(I_i, C_j) = \begin{cases} (p - m_p, -c + m_r) & \text{if } V_i > 0 \\ (0, 0) & \text{if } V_i < 0 \end{cases} \tag{3.12}$$

Proof To change the (C_i, C_j) action set to NE, (C_i, I_j), (I_i, C_j), and (I_i, I_j) should not be NE. That is

$$\begin{cases} p - c - m_p + m_r > p - m_p \\ p - c - m_p + m_r > -c + m_r \\ -c + m_r > 0 \end{cases} \tag{3.13a}$$

$$\Rightarrow p > m_p \ \& \ m_r > c. \tag{3.13b}$$

Proposition 3.4.2 *In a price system, the action set (C_i, C_j) is Pareto-Optimal iff $p > m_p$ & $m_r > c$.*

Proof Proposition 3.4.1 shows that iff $p > m_p$ & $m_r > c$, the action set (C_i, C_j) is the NE. Also, (C_i, C_j) is the best outcome in the system. Therefore, (C_i, C_j) is also Pareto-Optimal.

Proposition 3.4.2 indicates that price systems can provide effective cooperation incentives to the nodes.

Proposition 3.4.3 *Suppose a selfish node has dropped n_d packets and forwarded $n_r - n_d$ packets from its received n_r packets and that it has enough credits to pay the forwarding*

services for its generated n_g packets. If the selfish node manages to keep its credit amount above zero, the lower bound of its packet drop rate P_d is

$$P_d \geq \begin{cases} 1 - \alpha \cdot \frac{m_p}{m_r}, & \text{if } \lim_{n_r \to \infty} \frac{V}{n_r} = 0, \\ 1 - \alpha \cdot \frac{m_p}{m_r} + \frac{\beta}{m_r}, & \text{if } \lim_{n_r \to \infty, V \to \infty} \frac{V}{n_r} = \beta, \end{cases} \tag{3.14}$$

where $\alpha = \frac{n_g}{n_r}$ and V is the account value of the node.

Proof The selfish node has dropped n_d packets and forwarded $n_r - n_d$ packets from its received n_r packets, and it has enough credits to pay the forwarding services for its generated n_g packets. Therefore,

$$(n_r - n_d) \cdot m_r + V - n_g \cdot m_p \leq 0 \tag{3.15}$$

$$\Rightarrow P_d = \frac{n_d}{n_r} \geq \frac{n_r \cdot m_r + V - n_g \cdot m_p}{n_r \cdot m_r} \tag{3.16}$$

$$\Rightarrow \lim_{n_r \to \infty} P_d \geq \lim_{n_r \to \infty} \frac{n_r \cdot m_r + V - n_g \cdot m_p}{n_r \cdot m_r} \tag{3.17}$$

$$= \begin{cases} 1 - \alpha \cdot \frac{m_p}{m_r}, & \text{if } \lim_{n_r \to \infty} \frac{V}{n_r} = 0 \\ 1 - \alpha \cdot \frac{m_p}{m_r} + \frac{\beta}{m_r}, & \text{if } \lim_{n_r \to \infty} \frac{V}{n_r} = \beta. \end{cases} \tag{3.18}$$

Price systems detect selfish nodes by checking the node account value. Nodes with account values no more than zero are regarded as selfish nodes. Proposition 3.4.3 implies that price systems cannot detect some selfish nodes since they can drop packets while still keeping their account value above zero. Specifically, these systems cannot detect selfish nodes in three cases. First, the price system cannot detect selfish and wealthy nodes. Such a node has a considerable amount of credits (i.e., large V), which leads to large β, and

subsequently a large drop rate P_d according to Equation (3.14). Due to its large V, the selfish and wealthy node is not easily detected. Second, the price system cannot punish the selfish nodes in a high-traffic region where a node receives more packets than it generates (i.e., $n_g < n_r$). This condition leads to small α, which subsequently produces a large packet drop rate P_d according to Equation (3.14). Since the node consumes much fewer credits, it cannot be easily detected. On the other hand, the price system is unfair for nodes in a low-traffic region. Such a node may not be able to accumulate enough credits to buy forwarding services for its own packets although it is a cooperative node. Third, when the packet forwarding price is much smaller than the forwarding reward (i.e., $m_p \ll m_r$), P_d becomes very large according to Equation (3.14). Since a node's cooperative behavior enables it to buy several forwarding services, it can easily keep its account value above zero.

Proposition 3.4.4 *Given a price system with node packet drop probability q when its $V > 0$, its average packet drop probability is:*

$$P_d = \begin{cases} q, & \text{if } k \geq \alpha \\ \frac{k \cdot q}{k \cdot q + 1}, & \text{if } k < \alpha, \end{cases} \tag{3.19}$$

where $k = \frac{m_r}{m_p}$.

Proof The process of an account value change can be modeled as a Markov chain as shown in Figure 3.2 and Figure 3.3. Each cycle denotes the account state of a node with its account value. An labeled arrow between two states denotes the state transferring probability from one state to the other. We use s to denote a node's account state and $P_s(s)$ to denote the probability that the node is in state s.

Case 1 ($k \geq \alpha$): Figure 3.2 shows the Markov chain for the account states of a node when $k \geq \alpha$ (i.e., $m_r \cdot n_r \geq m_p \cdot n_g$). As shown by the right flowing arrows, when the node forwards a packet it gains m_r-m_p=$(k-1)m_p$ credits given $n_g = n_r$. As shown by

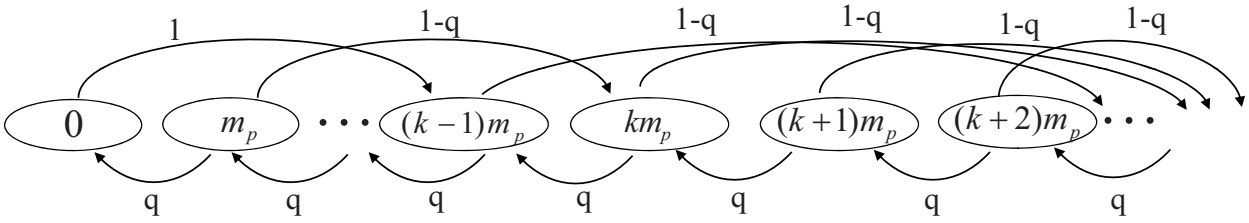


Figure 3.2: The Markov chain of the account states of a node when $k \geq \alpha$.

the left flowing arrows, when the node drops a packet it pays m_p credits for the forwarding service. When $s = 0$, the node has only one action choice – to be cooperative in order to buy services for its own packets. Therefore, the node jumps from state 0 to state $(k-1)m_p$ with probability 1. For other states, since $m_r \cdot n_r \geq m_p \cdot n_g$, i.e., the node has enough credits to pay its packet forwarding service, it can choose to drop or forward its received packets with probabilities q and $1-q$, respectively. Since the states in the Markov chain are infinite, i.e., $n_s \to \infty$ where n_s is the number of all states in the Markov chain, the probability that a node stays in state 0 is $\lim_{n_s \to \infty} P_s(0) = 0$. Because a node drops a packet with probability q only when $s \neq 0$, its average packet drop probability is

$$P_d = (1 - \lim_{n_s \to \infty} P_s(0)) \cdot q = q. \tag{3.20}$$

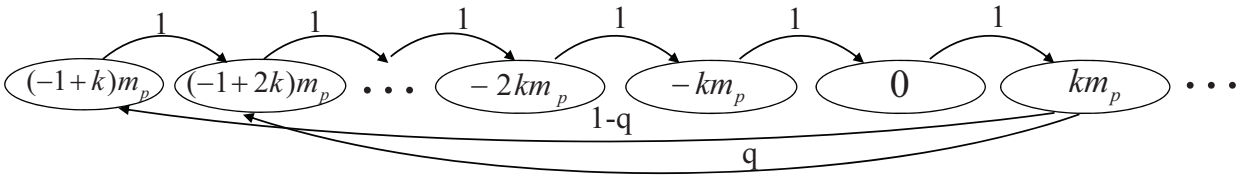


Figure 3.3: The Markov chain of the account states of a node when $k < \alpha$.

Case 2 ($k<\alpha$): Figure 3.3 shows the Markov chain of the account states of a node when $k<\alpha$ (i.e., $m_r \cdot n_r < m_p \cdot n_g$). It shows that states $\{(-1+k)m_p, (-1+2k)m_p, ..., -2km_p, -km_p, 0, km_p\}$ form a closed cycle. Thus, these states are called absorbing states [55] and the whole Markov chain can be reduced to the aborting states because a node cannot leave

the closed cycle once it stays in one of the absorbing states. As the left arrows show, when a node stays in the state km_p, it moves to state $(-1+k)m_p$ when it loses m_p credits by dropping a packet with probability q, and it moves to state $(-1+2k)m_p$ when it earns $(k-1)m_p$ credits by forwarding a packet with probability $(1-q)$. In other absorbing states, since their account values are not positive, the node has only one action choice – to be cooperative to increase its account value. Thus, as the right arrows show, a node forwards the packets with probability 1 in these states. Based on the global balance equations [55], we can obtain:

$$\begin{cases} P_s((-1+k)m_p) = q \cdot P_s(km_p) \\ P_s((-1+2k)m_p) = (1-q) \cdot P_s(km_p) + P_s((-1+k)m_p) \\ P_s((-1+3k)m_p) = P_s((-1+2k)m_p) = ... = P_s(0) = P_s(km_p) \\ P_s(km_p) + P_s(0) + ... + P_s((-1+2k)m_p) + P_s((-1+k)m_p) = 1 \end{cases} \tag{3.21a}$$

$$\Rightarrow P_s(km_p) = \frac{1}{q+\frac{1}{k}} = \frac{k}{k \cdot q + 1}. \tag{3.21b}$$

Since the node will drop packets only in state km_p as shown in the Markov chain, its packet drop rate is:

$$P_d = \frac{k \cdot q}{k \cdot q + 1}. \tag{3.22}$$

Repeated Games. In the price system, according to Proposition 3.4.1, when $p > m_p$ & $m_r > c$, the (C_i, C_j) strategy is both NE and Pareto-optimal. Therefore, in the repeated cooperation game with finitely many interactions, all nodes will choose (C_i, C_j) stably and continuously. Therefore, (C_i, C_j) in the repeated games of price system remains the NE and Pareto-optimal case.

3.5 The Design and Game Theory Model for the Integrated Reputation/Price System

One-interaction Game. A system that can effectively encourage the cooperation of the nodes in a one-interaction system should have two features: (1) strong incentives to encourage the nodes to be cooperative and (2) quick, effective detection of selfish nodes for punishment. The reputation system uses a reputation threshold to distinguish the selfish and cooperative nodes. However, it cannot provide strong incentive for cooperation. Though price systems can provide a strong incentive for node cooperation, they fail to provide an effective mechanism for detecting misbehaving nodes. We propose an integrated system that combines the reputation system and the price system. By integrating the misbehavior detection mechanism from the reputation system and the cooperation incentive mechanism from the price system, the integrated system can overcome the drawbacks of the individual systems.

In addition to the strategies of the individual reputation system and the price system, the integrated system offers additional strategies. Node i's packet forwarding price is determined from its reputation value by $m_p = \frac{a}{(R_i)^b}$, where a and b are constant parameters and b is used to control the increase/decrease speed of m_p based on R_i. Thus, a node with a higher reputation value may pay less for the packet forwarding service compared to lower reputation node. The reputation value R and account value V of each node are used to distinguish selfish nodes and cooperative nodes. The node with $V < 0$ or $R < T_R$ is regarded as a selfish node and its transmission requests will be rejected by other nodes.

Compared to the reputation system, the integrated system can effectively prevent some selfish nodes from keeping their reputation values just above the threshold value by selectively sending packets because the selfish nodes must pay more credits for packet forwarding, which will quickly deplete their credit account. Also, the system avoids discouraging the

cooperation of high-reputed nodes, since a higher reputed node can pay less for the packet forwarding. Compared to the price system, the integrated system can encourage wealthy nodes to always be cooperative in the packet forwarding because these nodes attempt to gain a higher reputation for a lower service price. The integrated system can also detect selfish and wealthy nodes in a high traffic region by reputation values, and encourage these nodes to be cooperative. If a node's reputation value is below the threshold ($R < T_R$) its transmission requests will be rejected by other nodes, regardless of its wealth. Therefore, the nodes stay cooperative for packet forwarding. Moreover, even in a low traffic region where a node has few chances to earn credits, a highly-reputed node can still have its packets forwarded because it pays a low price.

Table 3.6: Payoff matrix for the integrated system.

	Node j		
Node i		Cooperative	Non-cooperative
	Cooperative	$U(C_i, C_j)$	$U(C_i, I_j)$
	Non-cooperative	$U(I_i, C_j)$	(0, 0)

$$U(C_i, C_j) = (p - c + (m_r - \frac{m_p}{R_i}), p - c + (m_r - \frac{m_p}{R_j})). \tag{3.23}$$

$$U(C_i, I_j) = \begin{cases} (-c + m_r, p - \frac{m_p}{R_j}) & \text{if } V_j > 0 \ \& \ R_{I(j)} > T_R \\ (0, 0) & \text{if } V_j \leq 0 \parallel R_{I(i)} \leq T_R. \end{cases} \tag{3.24}$$

$$U(I_i, C_j) = \begin{cases} (p - \frac{m_p}{R_i}, -c + m_r) & \text{if } V_i > 0 \ \& \ R_{I(i)} > T_R \\ (0, 0) & \text{if } V_i \leq 0 \parallel R_{I(i)} \leq T_R. \end{cases} \tag{3.25}$$

Proposition 3.5.1 *In the integrated system, the action set (C_i, C_j) is both NE and Pareto-optimal if transmission cost c, current reputation value R_j and R_i, and packet forwarding reward m_r satisfy $m_r > c$ & $p > \frac{m_p}{R_i}$ & $p > \frac{m_p}{R_j}$.*

Proof To change the (C_i, C_j) strategy to be the NE and Pareto-optimal, the payoff values

of the integrated system should satisfy:

$$\begin{cases} p - c + m_r - \frac{m_p}{R_i} > p - \frac{m_p}{R_i} \\ p - c + m_r - \frac{m_p}{R_j} > p - \frac{m_p}{R_j} \\ p - \frac{m_p}{R_i} > 0 \\ p - \frac{m_p}{R_j} > 0 \end{cases}$$
$$\Rightarrow m_r > c \;\&\; p > \frac{m_p}{R_i} \;\&\; p > \frac{m_p}{R_j}.$$

Equations (3.23), (3.24) and (3.25) represent the payoffs of $U(C_i, C_j)$, $U(C_i, I_j)$, and $U(I_i, C_j)$. When the reputation value of a node is lower than threshold T_R, the node is regarded as a selfish node and punished. Therefore, node i needs to ensure $R_i > T_R$. That is, $p > \frac{m_p}{R_T} \rightarrow p > \frac{m_p}{R_i}$. As a result, the (C_i, C_j) strategy is always the NE and Pareto-optimal iff $m_r > c$ & $p > \frac{m_p}{R_T}$. Equations (3.23), (3.24) and (3.25) show that a high reputation value leads to a high payoff for cooperative behavior. Therefore, the integrated system can provide higher incentives than the price-based system for cooperative behavior, because the payoff earned by a cooperative behavior in the integrated system is higher than that in the price-based system.

In addition to providing incentive for higher node cooperation, the integrated system can also effectively detect selfish nodes by monitoring node reputation and account value. A selfish node is detected when its $R < T_R$ or $V < 0$. A silly selfish node is defined as the node that drops packets no matter whether its reputation value is below the reputation threshold T_R or not. A clever selfish node is defined as the node that selectively drops part of the receiving packets but keeps its reputation value above T_R. The wealthy and silly selfish node cannot be detected by the price system in a short time, but can be detected by the reputation component of the integrated system quickly when its reputation falls below T_R. Similarly, the selfish behaviors of the nodes with small packet forwarding requests cannot be detected by the price system, but can be detected by the integrated system when its

reputation falls below T_R. A wealthy and clever selfish node can avoid being detected in the reputation system. However, in the integrated system, the node's reputation drops quickly, and then its credits are quickly consumed as it always pays a very high price for packet forwarding services based on the price policy in the integrated system, leading to detection upon account starvation.

Repeated Games. In the one-interaction game of the integrated system, the (C_i, C_j) action set is both NE and Pareto-optimal iff $m_r > c$ & $p > c$. Thus, for a repeated cooperation game, each interacting node has no incentive to deviate from the (C_i, C_j) action set. Even if some nodes deviate from (C_i, C_j), the remaining nodes' payoff will not be reduced because (C_i, C_j) is the NE. Unlike IDS, nodes in the integrated system can always safely choose the cooperation strategy. Hence a MANET with the integrated system can always provide incentives for the nodes' cooperation. We define the *relative success rate* of a strategy as the rate of the total payoffs of nodes employing the strategy to the total payoffs of all nodes in the system. We also define *round* as a sequence of system interactions in which each pair of nodes have an interaction with each other. We use $f_{C/I}[t]$ to denote the percent of the nodes using strategy C or I in round t over all nodes.

Proposition 3.5.2 *The percent of the nodes adopting the cooperation strategy is*

$$f_C[t] = \frac{f_C[0]}{f_C[0] + f_I[0](\frac{U_i(I_i,C_j)+U_i(I_i,I_j)}{U_i(C_i,C_j)+U_i(C_i,I_j)})^{(t-1)}}. \tag{3.27}$$

Proof According to evolutionary game theory [73] and the property of linearity of expectation [55], we see that the percent of nodes adopting the cooperation strategy scales with the relative success rate of the cooperation strategy. According to the definition of the relative success rate, we obtain:

$$\frac{f_C[t]}{f_I[t]} = \frac{f_C[t-1]}{f_I[t-1]} \cdot \frac{U_i(C_i, C_j) + U_i(C_i, I_j)}{U_i(I_i, C_j) + U_i(I_i, I_j)} \tag{3.28}$$

$$\Rightarrow \frac{f_C[t]}{f_I[t]} = (\frac{U_i(C_i, C_j) + U_i(C_i, I_j)}{U_i(I_i, C_j) + U_i(I_i, I_j)})^t \frac{f_C[0]}{f_I[0]}. \tag{3.29}$$

Since $f_I[t] = 1 - f_C[t]$, we obtain:

$$f_C[t] = \frac{f_C[0]}{f_C[0] + f_I[0](\frac{U_i(I_i,C_j)+U_i(I_i,I_j)}{U_i(C_i,C_j)+U_i(C_i,I_j)})^{(t)}}. \tag{3.30}$$

Interestingly, in repeated games, if a selfish node changes to be cooperative in the next round $(t+1)$, the packet forwarding price decrease is:

$$\frac{m_p}{R(t+1)} - \frac{m_p}{R(t)} = \frac{(R(t+1) - R(t)) \cdot m_p}{R(t) \cdot R(t+1)}, \tag{3.31}$$

where $R(t)$ denotes the node's reputation at time t. That is, whether a node is high-reputed or low-reputed, the price for its packet forwarding requests always decreases in the next round if it is cooperative, and the price always increases in the next round if it drops packets. Therefore, the price policy in the integrated system can encourage both high-reputed and low-reputed nodes to be cooperative. Also, as Formula (3.31) shows that lower reputed nodes have more price reduced if they are cooperative in the next round, the lower reputed nodes receive higher incentives to be cooperative.

3.6 Summary

In this chapter, we analyzed the underlying cooperation incentives of defenseless, reputation and price systems through game theory. To overcome the observed drawbacks in each system, we proposed and analyzed an integrated system, which leverages the advantages of reputation and price systems. Analytical results show the higher performance of the

integrated system compared to the other two systems in terms of the effectiveness of the cooperation incentives and selfish node detection. In the next chapter, we propose an efficient reputation management system that is based on the integrated system proposed in this chapter.

Chapter 4

Scalability: A Hierarchical Account-aided Reputation Management System (ARM)

In Chapter 3, we theoretically studied the incentives provided by the reputation, price and integrated systems. Based on the theoretical results illustrated in Chapter 3, here we propose mechanisms to solve the second challenge in distributed wireless systems. That is, efficient cooperation incentives that are resource-efficient in use and maintenance. We propose a hierarchical Account-aided Reputation Management system (ARM) that can provide efficient cooperation incentives, which are resource-efficient in use and maintenance in large-scale MANETs.

Existing reputation systems and price systems are neither sufficiently efficient nor effective for a large scale MANETs. First, most current reputation systems lack efficient mechanisms to collect and propagate reputation information for large scale MANETs. Periodical information exchanges, keeping redundant reputations in each node, and broadcasting to query reputations consume significant resources and fail to achieve high scalability. Sec-

ond, reputation calculations based on partial local information, which may include false information, may result in an insufficiently accurate reputation evaluation to truly reflect node behaviors. Third, solely relying on a reputation system is not effective in thwarting uncooperative behaviors. Reputation systems provide equal treatment to trustworthy nodes with reputation values larger than the threshold. Thus, a node can be uncooperative for some time while maintaining its reputation value larger than the threshold, which leads to a suboptimal overall system performance.

Meanwhile, price systems also suffer a similar problem. First, the circulation of credits in the network requires a fair amount of computation and storage resources and increases traffic overhead. Second, these systems fail to provide a mechanism to measure the service quality offered by a node and lack effective methods to punish selfish and wealthy nodes (e.g., nodes that need few services) that sometimes drop others' packets. Third, cooperative nodes located in a low-traffic region receive few forwarding requests, and thus may not earn enough credits for their own requests, while nodes located in a high-traffic region have more chances to earn more credits than they actually need and thus may strategically drop some messages for their own benefit. Finally, the implementation of credits and virtual banks brings more complexity with additional requirements on transmission security. For example, since credits are stored at the head of a packet, which is transmitted through several nodes, preventing the credits from being stolen becomes a problem.

ARM builds a structure in a MANET to realize the integrated system proposed in Chapter 3 to provide stronger incentives than individual reputation and price systems. It selects low-mobility and trustworthy nodes as reputation managers (managers in short), builds them into a locality-aware distributed hash table (DHT) [88] infrastructure, and coordinately integrates the reputation and price systems through the infrastructure. DHTs are well-known for high scalability, efficiency and reliability, thus support scalable and efficient operations in ARM. The DHT marshals all reputation and transaction information for a

node into one manager, which calculates the reputation and increases/decreases credits for the node accordingly. Therefore, ARM can efficiently manage the reputation of nodes in the system with small overhead. It can also effectively avoid misbehaviors including reputation misreporting, reputation false accusation, and collusion.

Specifically, ARM consists of three components:

- *A locality-aware DHT infrastructure.* We study the requirements for creating such an infrastructure in a MANET and propose the construction and maintenance algorithms. The infrastructure efficiently collects all reputation and transaction information for a node for effective reputation and account management. Experimental results show that including the maintenance overhead for node mobility, ARM still generates much lower overhead than current reputation and prices systems.

- *Reputation management.* Relying on the collected global reputation information based on DHT, ARM effectively detects the false information and accurately calculates node reputation. Also, with the aid of the DHT, ARM reduces each node's burden for periodical information exchange and for storage and computing.

- *Reputation-adaptive account management.* ARM treats the nodes with different reputations accordingly and prevents nodes from gaining fraudulent benefits. Specifically, a higher-reputed node pays a lower price while a lower-reputed node pays a higher price for service. Also, a highly reputed node earns more credits than a lower reputed node for the same forwarding service. Using the DHT, ARM has no virtual credits circulating in the network, eliminating the implementation complexity and security concerns in virtual credits circulation.

The remainder of this chapter is organized as follows. Section 4.1 provides an overview of the ARM system. Section 4.2 introduces the algorithms to build a locality-aware DHT

Infrastructure. Section 4.3 and Section 4.4 illustrate how ARM manages the reputation values and prices of the nodes. Section 4.5 concludes this chapter. The experimental results are shown in Chapter 6.

4.1 An Overview of the ARM System

The resource managers constitute a locality-aware DHT, functioning as a backbone at the center of the MANET for efficient and stable operations of ARM. As shown in Figure 4.1, each normal mobile node has a *watchdog* [76, 69] to monitor and report the behavior of its neighbors to managers. The DHT helps marshal all reputation and transaction information of a given node in the system to a specific manager. The managers have two functions: reputation management and account management. Each manager calculates the reputations and increases/decreases the credits in the accounts of the mobile nodes for which it is responsible. Nodes with reputations either below the threshold or with deficit accounts are regarded as uncooperative. Managers notify mobile nodes about uncooperative nodes, which are then put onto their blacklists. The blacklisted nodes' forwarding requests are then ignored by others. Like price systems, ARM also requires the source node to pay the relay nodes for packet forwarding, but it eliminates the need for credit circulation in the network. Moreover, in ARM, a highly-reputed node pays less credits while a lower-reputed node pays more credits in a forwarding service transaction, thus effectively providing incentives for cooperation between nodes.

For example, when node n_1 looks for a path for packet transmissions, it broadcasts a path query message to the packet destination. When nodes n_2 and n_3 receive the query, they check whether n_1 is on their blacklists (step (1) in Figure 4.1). If so, they ignore n_1's query; otherwise, they respond to n_1. n_1 then forwards the packet along a discovered path consisting

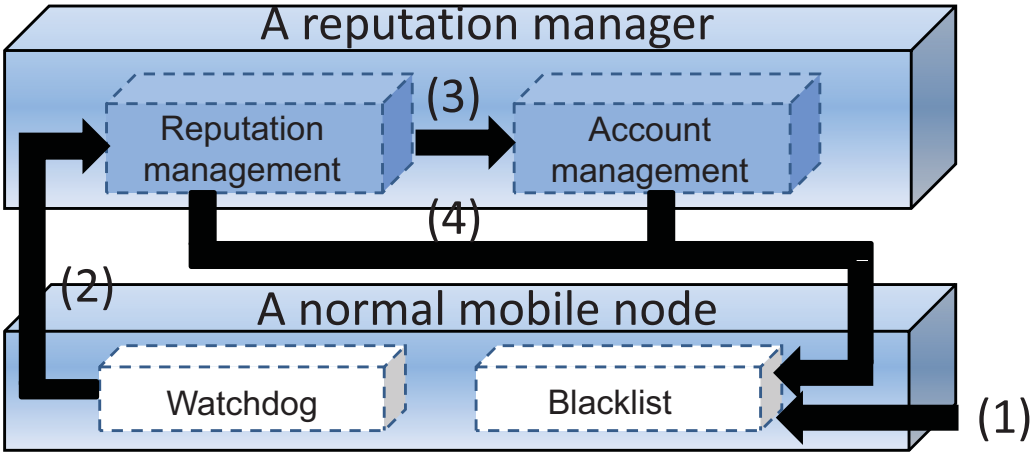


Figure 4.1: Overview of the ARM system.

of cooperative nodes including n_2 and n_3. In step (2), the neighbor nodes of communicating nodes n_2 and n_3 monitor the data transmission using their *watchdog*, and report the observed transmission rate to their closest managers. Relying on the DHT, the managers merge all reputation reports for n_2 and n_3 respectively and produce their global reputations. The DHT overlay supports efficient reputation information collection and querying. In step (3), ARM adds credits to the accounts of n_2 and n_3 and decreases the account of n_1. A higher reputation leads to more earned credits for n_2 and n_3, and lower service charges for n_1. In step (4), if the reputations of n_2 and n_3 are below a threshold or n_1 has a deficit account, managers inform all nodes in the network to place the uncooperative nodes on their blacklists.

Similar to [98], we assume that the movement of each node is independent and identically distributed (i.i.d.) in a square area with space length l. The number of smart phones is increasing daily with each typically having dual-mode: a low-power ad-hoc network interface (e.g., IEEE 802.11 interface) and a high-power infrastructure network interface (e.g., WLAN radio interface). Thus, we assume some mobile nodes in the MANET will have dual-mode interfaces. The network designers can initially deploy a number of peers in the network that serve as bootstrap manager nodes for DHT construction. The reputation messages exchanged between managers are of small size and delay tolerant compared to data messages.

Managers can use the low-power interface for data transmission and the high-power interface for reputation data transmission.

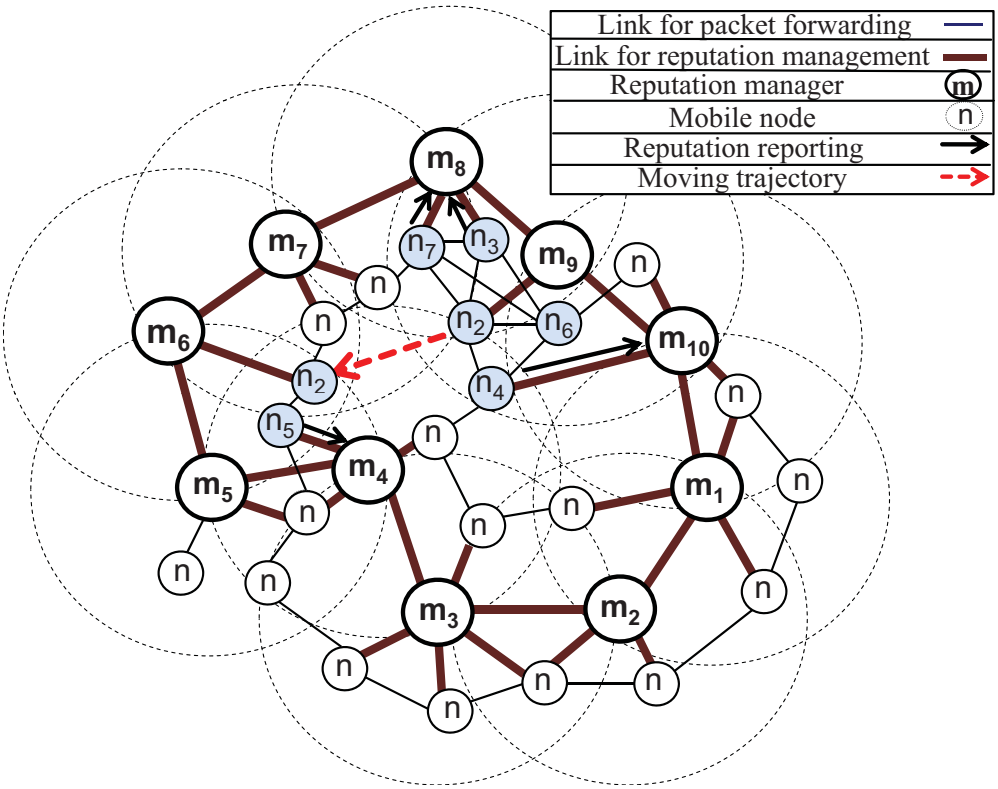


Figure 4.2: The ARM hierarchical structure.

4.2 Locality-aware DHT Infrastructure

Figure 4.2 illustrates the hierarchical structure of ARM. The higher level is a DHT network composed of managers (low-mobility and high-trustworthy nodes) and the lower level is composed of normal mobile nodes. A DHT network can partition ownership of a set of objects (e.g., files) among participating nodes and efficiently route messages to the unique owner of any given object. Each object or node is assigned an ID that is the hashed value of the object (e.g., file name) or node IP address using a consistent hash function [52]. An object is stored in a node whose ID equals or immediately succeeds the object's ID. The

DHT provides two main functions: `Insert(ID,object)` and `Lookup(ID)`, to store an object to a node responsible for the ID and to retrieve the object, respectively. The message for the two functions is forwarded based on the DHT routing algorithm. The DHT achieves $O(\log n)$ path length per lookup request by using $O(\log n)$ neighbors per node, where n is the number of nodes.

We leverage the Chord DHT network [88] as the ARM infrastructure for scalable and efficient reputation and account management. For each MANET, there is one DHT. ARM constructs a locality-aware DHT-based infrastructure where the logical proximity abstraction derived from ARM matches the physical proximity information. In this way, the packet routing path in the overlay is consistent with the packet routing path in the physical topology, which greatly reduces the physical routing distance and overhead. However, managers in MANETs are mobile while nodes in DHT networks are stable. Also, in a MANET, a node can only communicate with nodes within its transmission range. The limited transmission range of the node poses a challenge in building and maintaining a DHT in a mobile environment. Two questions naturally arise: (1) is it possible to form managers into a locality-aware DHT infrastructure in a MANET? (2) how is a DHT built and maintained in a mobile environment?

A Hamiltonian cycle graph is a graph in which a path can go through every vertex in the graph exactly once and return to the starting vertex [33]. A Chord DHT [88] with the ring topology is a representative of DHT overlays. A Chord, with the ring topology, is actually a Hamiltonian cycle because successor neighbor links connect all nodes to a circle. Therefore, to build a locality-aware DHT, the physical topology should also be a Hamiltonian cycle.

Proposition 4.2.1 *The transmission range of nodes should satisfy* $r \geq \frac{2}{\sqrt{2\pi}}l$ *in order to form a Hamiltonian cycle.*

Proof To guarantee that the nodes in a graph can form a Hamiltonian cycle, the number of neighbors of each node (i.e., connectivity degree) should satisfy $deg(v) \geq \frac{N}{2}$ where v denotes a vertex [33]. With the assumption of the i.i.d. movement, a node has the least connectivity degree when it moves to the corner of the square field. That is,

$$\frac{\pi r^2}{4l^2} N \geq \frac{N}{2} \Longrightarrow l \leq \frac{\sqrt{2\pi}}{2} r \Longrightarrow r \geq \frac{2}{\sqrt{2\pi}} l, \tag{4.1}$$

which shows the requirement for forming managers into a Hamiltonian cycle.

4.2.1 Locality-aware DHT Infrastructure Construction

Figure 4.3 shows an example of a physical topology and its corresponding logical topology in ARM. In a logical topology, the distance between node IDs represents their logical distance. To build managers into a locality-aware DHT infrastructure, we assign a sequence of consecutive DHT IDs to the managers along the path connecting all nodes in a cycle.

In a MANET, each node identifies its neighbors by sending "hello" messages. Thus, a node can infer the relative physical closeness of its neighbors by the communication latency. To assign IDs to managers, as shown in Figure 4.3, we first choose a trustworthy bootstrap manager (m_0) and assign it ID 0. Then, it chooses its physically closest node as its successor, and assigns it ID 1. The successor finds its successor and assigns it ID 2. The process is repeated until the bootstrap node is reached. At this time, a complete cycle is formed and all managers have been assigned numerically continuous IDs. The last node in the created path with ID $N-1$ must be in the transmission range of m_0, i.e., the successor of m_7 is m_0. Since only the physically close nodes can have sequential IDs, the constructed logical overlay topology is consistent with the physical topology of managers. Then, each manager builds a DHT routing table containing $\log N$ neighbors based on a DHT neighbor determination protocol using broadcasting.

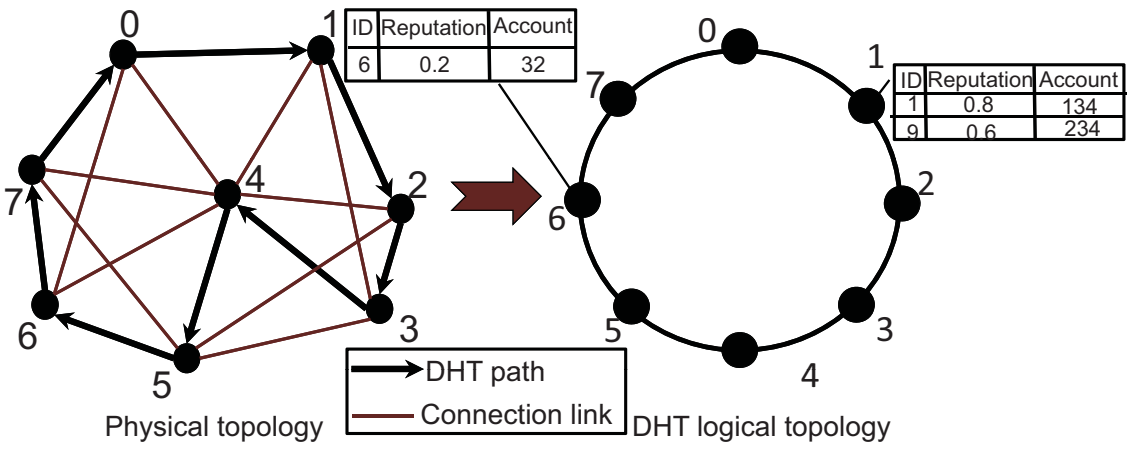


Figure 4.3: Construction of the DHT infrastructure.

4.2.2 Locality-aware DHT Infrastructure Maintenance

Proposition 4.2.2 *In ARM, the average time period a pair of neighbor managers stay within the transmission range of each other (i.e., connection duration) is* $\frac{r}{\overline{v}}$*, where* $\overline{v}$ *is the average relative speed of their movement.*

Proof Since the movement of each manager is i.i.d., if manager m_i is distance d away from manager m_j, the expected time period needed by m_i to move out of the transmission range of m_j is:

$$E(T) = \int_{2\pi}^{0} \frac{1}{2\pi} \frac{\sqrt{r^2 + d^2 - 2rdcos\theta}\cdot}{E(v)} \mathbf{d}(\theta) = \frac{r}{\overline{v}}. \tag{4.2}$$

Proposition 4.2.2 shows that the stability of the DHT infrastructure is primarily determined by the moving speed and transmission range of managers. To maintain the DHT structure in node mobility, managers must maintain connectivity with their neighbors to guarantee that they are sequentially connected from ID 0 to $N-1$. By regarding node movement as node departures followed by node joins, we can use the original DHT maintenance mechanism to maintain the ARM DHT infrastructure. However, it leads to a high maintenance overhead due to node mobility. We propose a lightweight DHT maintenance algorithm to deal with this mobility.

Each manager relies on the "hello" messages to check its connectivity with its successor and update the managers in its routing table. When manager m_i senses its link to its predecessor m_{i-1} is about to break based on the sensed transmission power, it notifies m_{i-1}. When manager m_{i-1} receives the notification or senses that its link to its successor m_i is about to break, it finds an alternative path that ends in m_k ($k > i-1$) and covers all managers with IDs $\in [i-1, k]$. The purpose of this operation is to maintain a complete DHT circle covering all managers with numerically continuous IDs. Since m_i moves in a local area to find the path with low overhead, m_{i-1} pings manager m_{i+j} ($j \geq 2$) sequentially by locally broadcasting a query message with $TTL = j$. That is, manager m_{i+2} is pinged first, then m_{i+3} is pinged, and so on. Each pinged manager replies to m_{i-1} with a message containing the routing path between them. Once the path covers ID$\in [i-1, k]$, m_{i-1} reassigns IDs to the managers in the detected path in sequence to maintain numerically continuous IDs in the cycle. If no path is found after half of the managers in the system are pinged, then m_{i-1} functions as a bootstrap manager for DHT reestablishment. For routing table maintenance, when a manager notices its routing table neighbor is not within its transmission range, it broadcasts a query message to find a new neighbor in that routing table entry.

As shown in Figure 4.4, when m_3 senses that its link to m_4 is about to break, it initializes a path querying process to find an alternate path covering all managers with ID$\in [3, k]$ starting from itself and ending with m_k. m_3 first pings m_5. If such a path cannot be found, m_3 pings m_6, then m_7, and so on. When an alternative path is discovered, the managers along the path will be assigned new consecutive IDs for a complete circle. After finding a new path that travels through manager m_i with ID 5 and m_j with ID 4, m_3 assigns m_i and m_j ID 4 and 5, respectively.

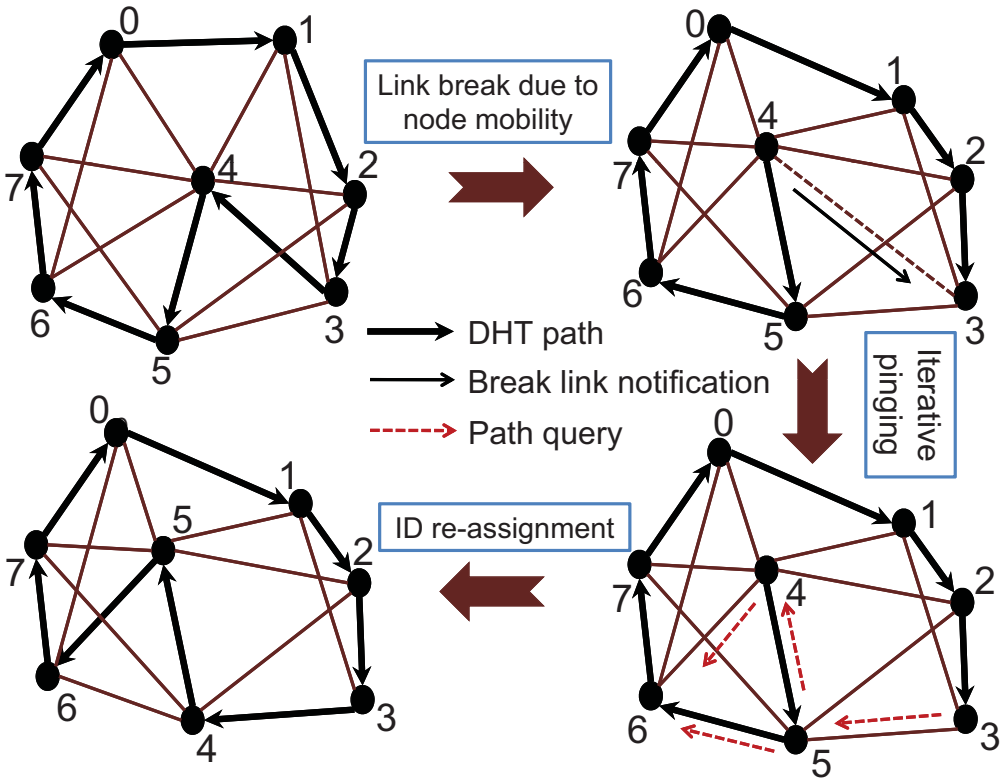


Figure 4.4: Maintenance of the DHT infrastructure.

4.2.3 DHT-based Information Collection and Querying

The DHT approach supports efficient and scalable information collection and querying in ARM. Each normal mobile node n_i has a virtual id=i, which is the consistent hash of its IP address. In a DHT, an object is stored in a node whose id equals or immediately succeeds the object's id. We call this node the object's owner manager. Nodes report the business and reputation information $(B + R)$ of the observed data forwarding behaviors of a specific node (n_i) to their nearest managers. Relying on the function `Insert(i,B+R)`, the managers marshal all information for n_i in the system to n_i's owner manager. The owner manager calculates n_i's reputation and increases/decreases the credits in its account. For example, in Figure 4.3, the information of the observed behavior of n_1 and n_9 is stored in m_1, which is responsible for their resource and account management. A node queries for the reputation of node n_i by sending the function `Lookup(i)` to its physically closest manager.

The query will be forwarded to the owner manager of node n_i relying on the DHT routing algorithm.

4.3 Reputation Management

In ARM, the reputation managers collect reputation information, calculate the global reputation, identify misbehaving nodes, and manage node accounts. ARM provides more accurate node reputation information for two reasons: it uses the global information rather than local partial information in reputation calculation and the large amount of global information makes it effective in detecting falsified, conspiratorial, and misreported information through deviation.

Neighbor Monitoring. ARM uses neighbor monitoring to observe the packet-forwarding behavior of nodes. Specifically, each observer uses a *watchdog* [76, 69] to keep track of the message forwarding behavior of its neighbors. The observer records the total number of packets that n_i has received from other nodes for forwarding (D_i^r), and the total number of packets that n_i has forwarded (D_i^f) during each time period T. Assume t_0 is the time instance that an observing node n_o joined the system. At each time instance $t_0 + kT$ ($k \in [1, 2, 3...]$), n_o calculates the observed reputation value of node n_i by $R_i^{n_o}=\frac{D_i^r}{D_i^f}$, reports this information to its closest manager, and resets D_i^r and D_i^f to zero. The manager then merges the collected reputations reported by nodes in its transmission range to the local reputation R_{l_i}.

Misreports Avoidance. When a node in a region experiences an adverse network condition, such as background interference due to traffic or thermal noise, the node's neighbors may also experience adverse network conditions. Thus, even though the nodes are cooperative, they are unable to transmit requested data. As a result, these nodes that mutually monitor each other report low R_l values for each other. In this case, the low R_l values are reported from nodes that are clustered together. Also, the interfering regions and the nodes

in the regions are changing constantly. ARM can easily solve this problem since it reports all reputations of a node in a region to a single manager. When a manager notices that all nodes in an area report low R_ls, it temporarily ignores the reports to reduce the uncertainty of the reported R_l to avoid punishing nodes for failing to forward packets due to adverse network conditions.

False Accusation Avoidance. Some misbehaving nodes may report a high reputation for an uncooperative node and a low reputation for a cooperative node. Since all observed reputations of a node in a region are collected into a manager and most nodes are benign, falsified reputations always deviate largely from most reported reputations. Thus, to reduce the effect of falsified reports, a manager filters the R_is that dramatically deviate from the average R_i. The deviation of $R_i^{n_o}$ reported by node n_o about node n_i is calculated as:

$$\Delta R_i^{n_o} = |R_i^{n_o} - \sum_{n_j \in \mathbf{n}} R_i^{n_j} / |\mathbf{n}||, \tag{4.3}$$

where $\mathbf{n}$ denotes the group of observers that report R_i to the manager during T, and $|\mathbf{n}|$ denotes the number of nodes in the group. ARM sets a threshold δ_l for the deviation and ignores $R_i^{n_o}$ satisfying $\Delta R_i^{n_o} > \delta_l$. The manager m_o then calculates the local reputation value of n_i in T denoted by $R_{l_i}^{m_o}$:

$$R_{l_i}^{m_o} = \sum_{n_o \in \tilde{\mathbf{n}}} R_i^{n_o} / |\tilde{\mathbf{n}}|, \tag{4.4}$$

where $\tilde{\mathbf{n}}$ denotes $\mathbf{n}$ after removing the deviated observed reputations. Then, the manager reports $R_{l_i}^{m_o}$ to n_i's owner manager using the function `Insert(i,`$R_{l_i}^{m_o}$`)`. According to equation (4.3), the expected value of δ is

$$E(\delta) = \left| \frac{a \cdot \overline{R}_{l_h} + b \cdot \overline{R}_{l_f}}{a+b} - \overline{R}_{l_f} \right| = \frac{a(\overline{R}_{l_h} - \overline{R}_{l_f})}{a+b}, \tag{4.5}$$

where $\overline{R}_{l_h}$ and $\overline{R}_{l_f}$ denote the expected values of honest reports and false reports respectively,

and a and b respectively denote the number of honest reports and the number of false reports in interval T.

Collusion Avoidance. The nodes in a region may collude to conspiratorially report node reputations to fraudulently increase their own reputations or decrease others' reputations. For example, the nodes in group A and group B are the nodes in the transmission range of m_k. The number of nodes in group B overwhelms group A. If the nodes in group B collude to report low R_i for n_i, then the justified reports from group A are ignored by m_k according to Equation (4.3). This problem can be resolved with another filtering process at the owner manager m_i that collects all $R_{l_i}^{m_o}$ from different managers m_o. Again, m_i computes the variance of $R_{l_i}^{m_o}$ based on Equation (4.6), and ignores $R_{l_i}^{m_o}$ with $\Delta R_{l_i}^{m_o} > \delta_g$. δ_g can be determined in the same way as δ_l:

$$\Delta R_{l_i}^{m_o} = |R_{l_i}^{m_o} - \sum_{m_j \in \mathbf{m}} R_{l_i}^{m_j} / |\mathbf{m}||, \tag{4.6}$$

where $\mathbf{m}$ is the number of managers that report R_{l_i}. Therefore, the global reputation of node n_i becomes:

$$R_{g_i} = \sum_{m_o \in \tilde{\mathbf{m}}} R_{l_i}^{m_o} / |\tilde{\mathbf{m}}|, \tag{4.7}$$

where $\tilde{\mathbf{m}}$ is the group of $\mathbf{m}$ after filtering.

For example, in Figure 4.2, nodes n_3, n_4, and n_7 monitor the transmissions of n_2. Nodes n_3 and n_7 report the observed reputation of n_2 to manager m_8 and n_4 reports its observed reputation of n_2 to m_{10}. Then, m_8 and m_{10} merge the reported reputations to a local reputation value of n_2 in its region, denoted by $R_{l_2}^{m_8}$ and $R_{l_2}^{m_{10}}$, and report the results to n_2's owner manager, m_2. Later, when n_2 moves close to n_5, n_5 will start monitoring the transmissions of n_2 and reporting the observed reputation of n_2 to its nearby manager m_4, which subsequently reports $R_{l_2}^{m_4}$ to manager m_2. Therefore, all local reputations of n_2 are marshaled to m_2, which then calculates the global reputation for n_2. Unlike most existing

reputation systems where a node calculates its neighbors' reputation values based on its local observations and cannot easily retrieve its new neighbor's previous reputation, ARM globally collects all R_{l_i} of node n_i at all times in all regions for global reputation calculation, leading to a more accurate reflection of n_i's trustworthiness over time. Also, global information (i.e., large data samples) makes it easier to detect false information.

When a node is suddenly out of power or suffers from channel congestion, it cannot offer service to others and thus has a low reputation despite not being selfish. It is unfair to punish such a cooperative node with a low reputation. On the other hand, it is difficult to identify the real reason for a low reputation. Therefore, ARM takes into account the old reputation when calculating the new reputation [11]. That is,

$$R_g^{new} = \alpha R_g^{old} + (1 - \alpha) R_g, \tag{4.8}$$

where R_g is the currently calculated reputation value for period T and α is a weight factor that is adaptive to the traffic load in the system. In a system with high traffic, a node is more likely to be either out of power or congested. Then, α should be set to a larger value. Therefore, we specify $\alpha = \bar{D}/\bar{C}$, where $\bar{D}$ is the average number of packets generated per second in the monitoring region and $\bar{C}$ is the expected channel capacity of the monitoring region.

ARM periodically decreases the reputations of the nodes whose $R_g > \beta\mathcal{T} + (1 - \beta)R_g^{max}$ $(\beta < 1)$ by:

$$R_g^{new} := \varphi R_g^{new} (\varphi < 1), \tag{4.9}$$

where R_g^{max} denotes the maximum global reputation, and β and φ are weight factors. The rationale behind this policy is that the reputation of a highly-reputed node will decrease over time if it does not receive a new rating from others. The low reputation subsequently increases the service price for message forwarding of the node (Section 4.4). Therefore, the

only way a node can enjoy a low price is to cooperate with other nodes, at all times. As with other reputation systems, ARM also sets a reputation threshold $\mathcal{T}$ to determine whether or not a node is selfish. In traditional reputation systems, smart selfish nodes may keep their reputation just above $\mathcal{T}$. Thus, they can sometimes drop packets while being regarded as reputed nodes. These nodes will be detected by the account management function in ARM. That is, if a node always generates packets rather than forwarding packets for others, it will eventually run out of credits and be detected as a selfish node.

Distributed Reputation Manager Auditing. Recall that reputation managers should be highly-reputed nodes with high reputation value. The high reputations of reputation managers only mean that they are willing to serve in packet forwarding, however, it does not necessarily mean that they would not misbehave in managing other nodes' reputations, e.g., modifying the reputation values of other nodes. A reputation manager may modify a node's reputation value and (or) account value in two situations. First, a reputation manager misreports the reputation of a node to its owner manager in the local reputation calculation. Second, the owner manager of a node modifies the reputation value and account value of the node in the global reputation calculation.

In the first situation, since the nodes in the transmission range of a reputation manager always change, the local reputation values of a node can be collected by several reputation managers in an interval T. After these managers report the collected reputation values of a node to its owner manager, the owner manager can detect the misbehaviors of the malevolent reputation mangers using the collusion avoidance method introduced in Section 4.3.

In the second situation, as the owner reputation manager calculates the final reputation value and manages the account value for a node, if the manager modifies the reputation value, no other nodes can detect it. To handle this problem, we use redundant reputation managers for each node. Specifically, we set c different consistent hash functions. When a

manager reports the local reputation of a node to its owner managers, it uses the c consistent hash functions to generate c virtual ids. Then, it uses `Insert(id,B+R)` to report the reputation to the owner managers of the node. When a node inquires the reputation value of node n_i from reputation managers, it also uses the c consistent hash functions on n_i's IP address to generate c virtual ids. Then, it executes `Lookup(id)` to retrieve the values. The node first calculates the average of the c returned values. The reputation managers whose returned reputation values deviate from the average value for a certain threshold δ_a are considered as malevolent managers. Then, the node regards the average value of the reputation values from the non-malevolent managers as n_i's global reputation value. The node also reports the suspicious malevolent manager to other $c-1$ managers. The managers periodically exchange their received misbehavior reports, and dismiss the manager who has been reported as a malevolent manager after checking the reputation values managed by the owner manager by executing `Lookup(id)`.

In this case, a highly-reputed node will be selected to join the DHT to replace the dismissed manager. To select a new reputation manager, the reputation manager with DHT ID=0 initially selects a node with the highest reputation among the normal nodes it manages. Then, the manager transfers a token, *TOKEN(Reputation value (RV_t), Reputation manager ID (ID_0))*, to the reputation manager with DHT ID=1. If the reputation manager has normal nodes with reputation value (RV) higher than RV_t in the TOKEN, it replaces RV_t with RV and replaces ID_0 with ID_1 in the token and passes the token to its successor reputation manager. The process continues until the reputation manager with ID=0 in the token receives the token. Then the reputation manager informs the node with reputation value RV_t to be the reputation manager. From this point, the locality-aware DHT infrastructure maintenance algorithm in Section 4.2.2 is used to maintain the locality of the DHT infrastructure.

4.4 Reputation-adaptive Account Management

ARM has an account management function to avoid equal treatment of highly-reputed nodes in different reputation levels to effectively provide cooperation incentives and deter selfish behaviors. ARM assigns each newly joined node with an initial number of credits denoted by $A(0)$. The owner managers of nodes maintain their accounts and transparently increase and decrease the credits in the accounts of forwarding service providers and receivers, respectively. Thus, as opposed to previous price systems, ARM's account management does not require credit circulation in the network, reducing transmission overhead, system complexity and improving communication security.

In previous price systems [66, 57], the credits a node earns or pays, equals the product of the unit price and the absolute number of packets forwarded (*absolute method* in short). Cooperative nodes in a region with low traffic may not earn enough credits for their transmission needs, and nodes in a region with high traffic or without many transmission service needs can be uncooperative without being punished. To deal with these problems, rather than relying on the absolute number, ARM determines the credits earned by a node based on the percent of forwarded packets among its received packets. Notice that R_g is exactly the percentage in ARM; we use it directly for the calculation of earned credits. Specifically, node n_i's owner manager increases its account every period T by

$$P_e = p_r R_{g_i}, \tag{4.10}$$

where p_r is a constant credit rewarding factor. We call this method the *relative method*. The relative method is advantageous because: (1) managers can directly use the latest reported reputation for account calculation instead of taking extra effort to record packet forwarding activities between nodes, reducing transmission overhead, and (2) it awards nodes fairly according to the cooperative degree of node behavior.

Proposition 4.4.1 *For cooperative behavior rewarding, the relative method provides nodes fairer treatment than the absolute method.*

Proof We use q_l and q_h (q_h>q_l) to denote the percent of the time period T used for packet transmissions in a relay node in low-traffic and high-traffic regions, respectively. In the absolute method, we use p_a to denote the amount of awarded credits per packet. Suppose λ is the average packet generation rate of the source, during time period t, the cooperative relay node gains $(q_h - q_l)tp_a \cdot \lambda R_g$ more credits in the high-traffic region than in the low-traffic region. Using the relative method, whether the relay node is in a low-traffic region or a high-traffic region, it always earns $\frac{t}{T}p_r R_g$.

To foster the cooperation incentives, ARM connects the forwarding service cost per packet p_c of a node to its reputation, so that higher-reputed nodes receive more credits while lower-reputed nodes receive fewer credits for offering the same forwarding service. The p_c of n_i, denoted by p_{c_i}, is calculated by:

$$p_{c_i} = \frac{\gamma}{R_{g_i}^{new}}, \tag{4.11}$$

where γ is a weight.

When an observing node n_o notices that N_{p_i} packets of node n_i have been transmitted by others during period T, it reports this business information B_i to its nearest manager along with R_i. By the DHT function `Insert(i,` $B_i + R_i$`)`, the manager forwards the information to n_i's owner manager m_i, which then deducts $p_{c_i}N_{p_i}$ credits from n_i's account. Therefore, the account of node n_i at time $t_0 + kT$ ($k \in [1, 2, 3...]$) is:

$$A(t) = A(0) - \sum_{t=t_0}^{t_0+kT} (p_{c_i}(t) \cdot N_{p_i}(t) - p_r \cdot R_{g_i}). \tag{4.12}$$

When the account of node n_i is negative, managers notify all nodes to place node n_i in their blacklists.

Proposition 4.4.2 *ARM exponentially increases the credits of a node while it is cooperative and exponentially decreases the credits of a node while it is uncooperative.*

Proof Suppose a node's R_g stays approximately constant during a time period T when it sticks to an certain action strategy. We use $R_g(t)$ to denote the reputation of a node at an arbitrary time instance $t = kT$ ($k \in [0, 1, \ldots, m]$) during time period mT. $R_g(t+T)$ and $R_g(t)$ correspond to R_g^{new} and R_g^{old} in the $(k+1)^{th}$ time period.

From Equation (4.8), we can determine that:

$$R_g(t+T) - R_g = \alpha \cdot (R_g(t) - R_g), \tag{4.13}$$

$$\Rightarrow R_g(t) = \alpha^{\frac{t}{T}}(R_g(0) - R_g) + R_g. \tag{4.14}$$

Based on Formulas (4.11) and (4.12), after time t, a node's account is:

$$A = A(0) - \sum_{t=T}^{kT}(p_c(t) \cdot N_p - p_r R_g) \tag{4.15a}$$

$$> A(0) - \int_{T}^{kT+T}(p_c(t) \cdot N_p - p_r R_g) \cdot \mathbf{d}(t), \tag{4.15b}$$

$$= \begin{cases} A(0) - \frac{\gamma N_p(1-\alpha^{-k})\cdot T}{(R_g - R_g(0))\cdot \ln\alpha \cdot \alpha} + p_r \cdot T \cdot R_g \cdot k & \text{if } R_g \neq R_g(0) \\ A(0) - \frac{\gamma N_p \cdot k \cdot T}{R_g} + p_r \cdot T \cdot R_g \cdot k & \text{if } R_g = R_g(0). \end{cases} \tag{4.15c}$$

Because $\alpha^{-k} > 1$ and $\ln\alpha < 0$, when $R_g < R_g(0)$, the account exponentially decreases with k; when $R_g > R_g(0)$, the account exponentially increases with k; and when $R_g = R_g(0)$, the account decreases linearly with k.

From Proposition 4.4.2, we can deduce that to ensure a selfish node will finally run out of the credits if it manipulates its reputation just above the threshold $T_R = R_g$, we must

ensure:

$$\gamma > \begin{cases} \frac{(R_g - R_g(0)) \ln \alpha \cdot \alpha \cdot (A(0) + Tkp_r R_g)}{(1-\alpha^{-k}) N_p T} & \text{if } R_g \neq R_g(0) \\ \frac{(A(0) + Tp_r R_g k) \cdot R_g}{T N_p k} & \text{if } R_g = R_g(0). \end{cases} \tag{4.16}$$

4.5 Summary

In this chapter, we propose a hierarchical Account-aided Reputation Management system (ARM) to efficiently and effectively deter node selfish behaviors and provide cooperation incentives. ARM builds an underlying locality-aware DHT infrastructure to efficiently collect global reputation information in the entire system for node reputation evaluation, which avoids a periodical message exchange, reduces information redundancy, and more accurately reflects a node's trustworthiness. ARM has functions of reputation management and account management, the integration of which fosters the cooperation incentives and non-cooperation deterrence. ARM can detect the uncooperative nodes that gain fraudulent benefits while still being considered as trustworthy in previous reputation and price systems. Also, it can effectively identify falsified, conspiratorial and misreported information so as to provide accurate node reputations that truly reflect node behaviors. Based on the scalability and trustworthy reputation management system as ARM, we present an efficient routing algorithm for delay tolerant network in next chapter.

Chapter 5

Efficiency: A Social Network and Utility based Distributed Multi-copy Routing Protocol (SEDUM)

In Chapter 4, we proposed a hierarchical Account-aided Reputation Management system (ARM) to efficiently and effectively deter node selfish behaviors and provide cooperation incentives. To handle the third challenge in distributed wireless systems, we propose an efficient routing algorithm to deliver messages with less overhead and lower delay. Specifically, we propose a Social nEtwork and utility based DistribUted Multi-copy routing protocol (SEDUM) for Delay Tolerant Networks (DTNs) that fully exploits node movement patterns to increase routing throughput and decrease routing delay. We assume that based on the ARM reputation management system, the nodes are cooperative in packet forwarding.

Routing methods specifically for DTNs have been widely studied in recent years. One group of routing methods use flooding [75, 90, 94, 97] to enable a message to opportunistically meet its destination node. Despite their high robustness and low transmission

delay, flooding-based routing methods require high energy, bandwidth, and memory space that are precious resources in wireless networks. Under high traffic loads, these methods suffer from severe resource contention and message dropping, which significantly degrade their efficiency. The other group of methods use single-copy routing, such as direct routing [87] and probabilistic (i.e., predicted) routing [47, 9, 27, 17, 71]. In direct routing, a source node spreads messages to several mobile nodes, which keep messages until they meet the destination node. In probabilistic routing, the messages are forwarded to mobile nodes that have higher probabilities of meeting the destination node as measured by the contact frequency utility. Although the single-copy methods save node resources and produce lower transmission overhead, they are likely to suffer from severe transmission delay if a suboptimal forwarding node (i.e., a node not in the shortest S-D path) is chosen.

In many DTN applications, such as mobile sensor networks [75], vehicular networks [39], and networks formed by mobile phone holders, the movements of mobile devices exhibit certain patterns as the devices in these scenarios are the extension of the hosts (i.e., human or animals), which normally exhibit certain movement routines [27]. Some nodes, such as home neighbors and colleagues, have a high probability of meeting with each other and staying close for a long time. This attribute of a movement pattern is called *colocation* [27] in a network. Some nodes, such as students on a campus, meet each other with high frequency but a short meeting time. This attribute of the movement pattern is called *familiar stranger* in a social network [77].

The movement pattern of nodes can be leveraged to assist a node in finding a relay node with a high probability of successfully sending data to the destination. Intuitively, familiar strangers normally have high contact frequency, but cannot guarantee the transmission of a large number of messages during a contact due to limited contact time. On the other hand, colocation nodes may have low contact frequency, but they have a long meeting time during each contact, in which a large number of messages can be transmitted. A few

routing protocols have been proposed to explore social communities in social networks for data routing in DTNs [41, 60, 35, 28, 30, 42, 34, 25, 24] by aggregating contacts among nodes in the past to a social graph, which may not be applicable to a large network with dynamically changing network size and node movement pattern.

In this chapter, we provide answers to the following questions:

- Is there a metric that can capture the *colocation* and *familiar stranger* features of the node's movement to assist the message routing in a DTN?
- How to build a routing algorithm that can leverage the advantages of the single-copy routing and flooding?

Specifically, SEDUM consists of three distinguishing components.

- *Duration utility based distributed routing.* We propose a *duration utility*, which is the ratio of total contact duration between two nodes over a time period T. A high duration utility between two nodes indicates a high message transmission throughput between them. This utility can fully capture the colocation and familiar stranger attributes of the node movement pattern in the social network. Forwarding messages to nodes that have higher duration utilities with destinations enhances routing throughput and decreases routing delay.
- *Efficient multi-copy routing.* Rather than relying on either flooding or single-copy routing, SEDUM uses multi-copy routing to achieve a tradeoff between routing delay and overhead. It uses the optimal tree replication algorithm to enable a node to quickly replicate a number of copies to other nodes while moving. We theoretically analyze the efficiency of this replication algorithm and the influence of the replication delay on the routing delay. We also build a Markov chain to model the replication process, which helps discover the minimum message copies necessary to achieve a desired routing delay.

- *Effective buffer management.* The buffer management mechanism gives longer-lifetime messages a higher priority to be sent from buffers, thus reducing the system's total transmission latency. It also gives higher-utility messages higher priority to remain in buffers when there is congested, thus increasing the system's total throughput. Further, it quickly deletes the replicas of delivered messages to releases buffer congestion.

The rest of this chapter is structured as follows. Section 5.1 theoretically analyzes why a duration utility is better than a contact utility for enhancing throughput. Section 5.2 explains the SEDUM routing protocol in detail. Section 5.3 provides a theoretical analysis of SEDUM. Finally, Section 5.4 concludes the chapter.

5.1 Why Duration Utility Is Better Than Frequency Utility

5.1.1 Frequency Utility

In DTN routing, the utility of a node is a measure of the contribution of the node to enhance a routing metric such as throughput or delay [7]. The contact frequency utility is widely used for probabilistic routing in DTNs. Node n_i's contact frequency utility to node n_j is defined as the ratio of the number of contacts between n_i and n_j in a time period. In frequency utility based routing, a node chooses its neighbor with the highest utility to the destination as the next hop for high routing successful rate and throughput.

5.1.2 Factors Affecting the Successful Transmission

We consider one message as a basic unit for the transmission between two nodes. If the link between two contacting nodes breaks before a message is completely transmitted,

the message transmission fails. In a multi-copy routing protocol, each copy is transmitted independently. Suppose that each message can have N_c copies and each of the copies can be successfully transmitted from the source to the destination with probability $P_{(S,D)}$. Then, the probability that at least one copy is sent to the destination node (P) is:

$$P = 1 - (1 - P_{(S,D)})^{N_c}. \tag{5.1}$$

Equation (5.1) shows that a larger $P_{(S,D)}$ and a larger N_c lead to a higher P_s. However, a larger N_c generates higher transmission overhead. Later on, we prove that increasing a large N_c leads to a linear increase in transmission overhead but a negligible delay decrease (Theorem 5.3.1). Therefore, we aim to increase the value of $P_{(S,D)}$. $P_{(S,D)} = \prod P_{(i,j)}$, where $P_{(i,j)}$ is the probability of successful transmission between two neighboring nodes n_i and n_j in a routing path. A large $P_{(i,j)}$ leads to a large $P_{(S,D)}$, and ultimately a large P_s. Next, we will find the factors that should be considered in order to increase $P_{(i,j)}$.

We use α to denote the smallest contact duration between two nodes at one contact. Specifically, $\alpha = \frac{R}{2v_{max}}$, where R is the transmission range of the mobile nodes and v_{max} is the maximum moving speed of a mobile node. We use f to denote the contact frequency between two nodes. Then, the two nodes have fT contacts during the time interval T. We use $P_{(i,j)}(fT = 1)$ to denote $P_{(i,j)}$ when $fT = 1$, and use $P_{(i,j)}(fT > 1)$ to denote $P_{(i,j)}$ when $fT > 1$.

Theorem 5.1.1 *The probability of a successful message transmission between two neighboring nodes, $P_{(i,j)}$, during a time interval T is:*

$$\begin{cases} P_{(i,j)}(fT = 1) = (\dfrac{\alpha \cdot w}{s})^{\beta} \\ P_{(i,j)}(fT > 1) = 1 - \left(1 - (\dfrac{\alpha \cdot w}{s})^{\beta}\right)^{fT} \quad (\beta > 0), \end{cases} \tag{5.2}$$

where w is the transmission rate of a node, s is the size of a message, and β is a constant parameter.

Proof Chaintreau *et al.* [22] indicated that the communication time of one contact between two persons conforms to a power-law distribution, and given α and β, the distribution of the contact time period t can be modeled by:

$$p(t) = \frac{\beta \cdot \alpha^{\beta}}{t^{\beta+1}} \quad (0 < \alpha < t < \infty, \beta > 0). \tag{5.3}$$

As the amount of transmission traffic during time t is $W = wt$, Equation (5.3) can be transformed to:

$$p(W) = \frac{1}{w} \frac{\beta \cdot \alpha^{\beta}}{(\frac{W}{w})^{\beta+1}}. \tag{5.4}$$

Note that, for two contacting nodes, only when their communication capacity in the contact is larger than the message size ($W > s$), will the message be transmitted successfully. Therefore, based on Equation (5.4), we obtain:

$$P_{(i,j)}(fT = 1) = P_{(i,j)}(W > s) = \int_{s}^{\infty} p(W) \cdot dW = (\frac{\alpha \cdot w}{s})^{\beta}, \tag{5.5}$$

$$P_{(i,j)}(fT > 1) = 1 - \left(1 - (\frac{\alpha \cdot w}{s})^{\beta}\right)^{fT}. \tag{5.6}$$

From Formula (5.6), we can see that the success probability for a message is determined by both $\alpha \cdot w$ and f. A large frequency utility cannot ensure a high transmission success probability if $\alpha \cdot w$ is very small. Also, a small frequency utility does not necessarily indicate a small transmission success probability if $\alpha \cdot w$ is very large. Therefore, frequency utility is not the only factor that affects the transmission throughput between two nodes. The frequency utility works well when the nodes in a network have a medium mobility rate (i.e., medium or large α), meaning a node can completely forward a message to the des-

tination when they meet. However, when nodes have high mobility rates (i.e., small α), the communication time during one contact between two nodes is short. Then, it is likely that the link between two nodes breaks during the message transmission process, leading to message transmission failures.

5.1.3 Duration Utility

Therefore, the contact frequency utility f cannot guarantee high communication capacity and throughput of a DTN, and we need to have a new utility that can reflect both $\alpha \cdot w$ and f. Since w of a given pair of nodes is determined, to reflect α, we propose a duration utility between nodes n_i and n_j as

$$U_{(i,j)} = \big(\sum_{k=1}^{fT} t_{(i,j)}(k)\big)/T, \tag{5.7}$$

where $t_{(i,j)}(k)$ is the encounter duration of the k^{th} encounter.

Theorem 5.1.2 *A duration utility can reflect the transmission capacity between a pair of nodes with higher accuracy than a contact frequency utility.*

A large duration utility indicates either a large $a \cdot w$, a large fT, or both. Therefore, the duration utility can more accurately reflect the transmission success probability P than the contact frequency utility, especially in a DTN with high-mobility nodes and large messages.

5.2 Duration Utility Based Distributed Multi-Copy Routing Protocol

In this section, we present the details of the duration utility based distributed multi-copy routing in SEDUM.

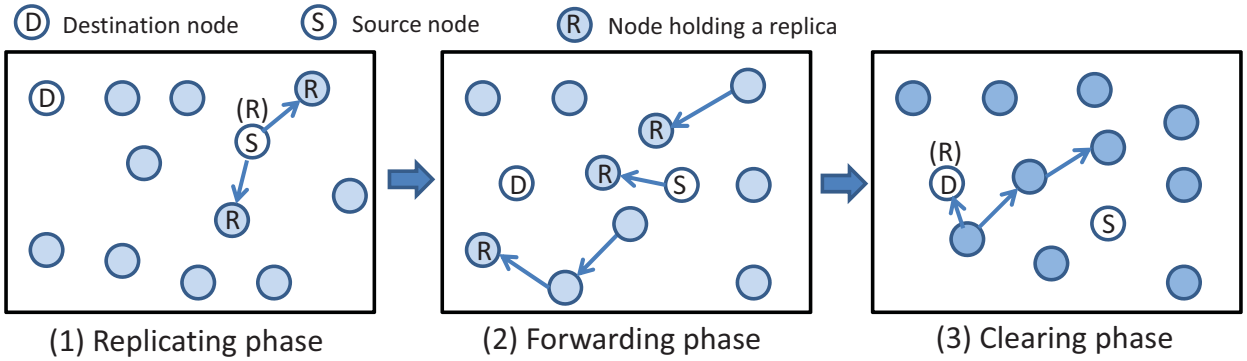


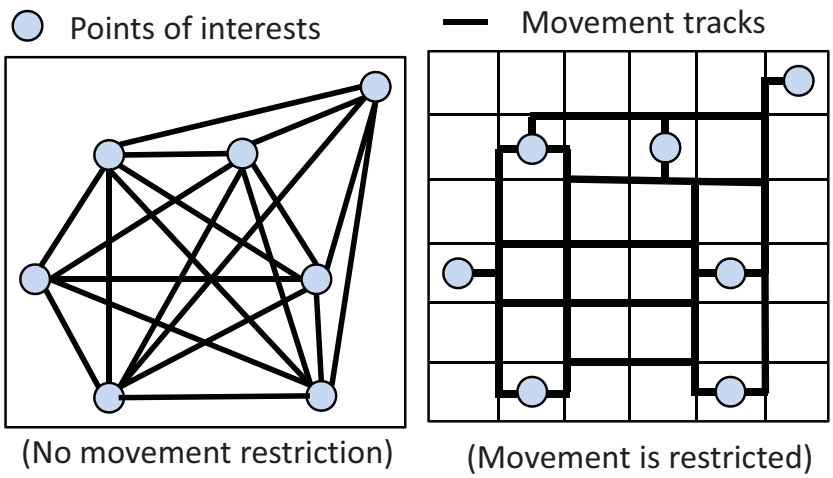
Figure 5.1: An example of message routing in SEDUM.

To route a message in SEDUM, a source node quickly spreads a number of message replicas to a number of nodes that it meets. The message replicas are transmitted simultaneously throughout the entire network until one copy reaches the destination node. Specifically, SEDUM can be generally divided into three phases: *Replicating phase*, *Forwarding phase*, and *Clearing phase* as shown in Figure 5.1.

(1) Replicating phase: Every message originating at a source node is initially replicated to a number of different meeting nodes.

(2) Forwarding phase: Each node in the system maintains a utility table recording its duration utilities to other nodes. A node always forwards a message to another node with a higher utility to the destination. This process is repeated until one copy of the message arrives at the destination node. A node is notified about the message delivery in the clearing phase.

(3) Clearing phase: After a message transmission is completed, the destination node notifies the nodes in the system to discard the replicas of the delivered message by sending a delivered message list. The lists are exchanged between two nodes when they meet.



(a) Basic community model (b) Manhattan community model

Figure 5.2: Community models.

5.2.1 Node Movement Models

The traditional popular node movement models such as the random walk model and the random way-point model [72] assume that the nodes are i.i.d. in the system and that each node independently moves with equal frequency to every network location. Numerous recent studies on human traces (e.g., university campuses and conferences) demonstrate that these two models rarely hold true in real-life situations where mobile devices are controlled by humans [87]. In this case, mobile node movement is based on human decisions and social behaviors.

Chaintreau *et al.* [22] studied the data transfer opportunities between wireless devices carried by humans. They observed that the inner-community contact duration of nodes follows a heavy-tailed distribution over the range from 10 minutes to 1 day and that the inter-community contact time distribution can be modeled as the power-law distribution. This is because in daily life, people spend most of their time with others in the same community, such as colleagues, parents, roommates, and etc. Although the number of persons we may meet daily is large, most meetings have a very short communication time.

Therefore, we consider a more realistic mobility model called the community model, which captures the movement patterns of the human nodes in the social network [22]. The community model has many communities, such as home, gathering places, and working places. In the model, every person has his/her movement routine. That is, when a person is at one place, he/she will go to some places with higher probability or go to other places with lower probability. For example, if a node is at its home community, it will go to a gathering place (e.g., mall or park) with a high probability. If a node is at a gathering place, it is very likely that its next destination is home.

Figure 5.2 shows a Basic community model [59] and a Manhattan community model [48]. In the former, there are no movement path restrictions on nodes. The nodes randomly select a speed and move to the destination directly. Though the tracks of the nodes' movements are stochastic, it is impossible to build enough roads or paths to directly connect every pair of destinations in real life. Thus, this model is not very suitable to simulate human behavior in a practical situation. The Manhattan community model [48] uses a grid road topology, where the mobile nodes move along the grid in horizontal and vertical directions akin to roads. This model is more realistic because we can only travel on the roads or paths that connect different places in the real world.

5.2.2 Duration Utility Calculation

In this section, we introduce a method for calculating the duration utility, which considers both contact frequency and duration between two nodes in a time interval T. Each node n_i periodically records the accumulated contact duration with the individual nodes it has met in a time period T.

Node n_i can either directly send a message to n_j or send a message to n_j though n_k, i.e., $n_i \rightarrow n_k \rightarrow n_j$. In this case, we say n_i has an *indirect* duration utility $\widetilde{U}_{(i,j)}$ with node n_j. The indirect duration utility between n_i and n_j through n_k is calculated using the

transitive principle:

$$\widetilde{U}_{(i,j)} = \widehat{U}_{(i,k)} * \widehat{U}_{(k,j)}. \tag{5.8}$$

Finally, the duration utility $U_{(i,j)}$ equals:

$$U_{(i,j)} = \max(\widehat{U}_{(i,j)}, \max_{k \in N}(\widetilde{U}_{(i,j)})), \tag{5.9}$$

where N is the set of all nodes in the network. That is, the duration utility between two nodes is the maximum of their direct utility and indirect utility. Thus, two nodes with a low contact frequency still have a high duration utility if they have a long meeting time. Even if two nodes have a low direct duration utility, if both have high duration utilities to a common node, they can still have a high utility to each other by forwarding messages through the common node.

Based on Formula (5.9), a node periodically calculates its delivery utility with all other nodes. A node's movement pattern may change in a social network due to reasons such as an office change, vacations, and etc. So that the duration utility more accurately reflects the current communication capacity, a node periodically updates the expected duration utility every T by considering both the historical utility and the current utility:

$$U_{(i,j)_{new}} = \gamma U_{(i,j)} + (1-\gamma) U_{(i,j)_{old}}, \gamma \in (0,1), \tag{5.10}$$

where γ is a weight constant and $U_{(i,j)_{new}}$ and $U_{(i,j)_{old}}$ respectively denote the utility of the new and old time intervals. The system with high dynamic changes in the pattern of movement can set γ to a large value to give more weight to the newly calculated utility value to reflect the dynamic change of the overall utility value.

Each node has a *utility table* to store its utilities with other nodes. Figure 5.3 shows an example of the utility table of node n_i in SEDUM. It records the duration utility of n_i

Routing table

Node	Delivery utility	Relay
n_1	0.7	n_2
n_2	0.8	N/A
n_3	0.7	N/A
n_7	0.1	N/A
n_9	0.6	n_1

Figure 5.3: An example of the utility table of a node.

with all the nodes that n_i has met. The "Relay" in the table indicates whether the duration utility between n_i and n_j is a direct utility or an indirect utility. In this column, "N/A" means direct utility and node "n_k" means the utility is indirectly calculated through n_k. For example, the direct utility of n_i and n_2 is 0.8, and the indirect utility of n_i and n_1 is 0.7 calculated through n_2.

In routing, either a source node or a relay node forwards a message to the neighbor that has the highest duration utility to the destination. As Figure 5.3 shows, the utility of n_i to n_2 is 0.8 and the utility of n_i to n_9 is 0.6 through n_i. If node n_i is asked to transmit a message to node n_2, n_i holds the message until meeting n_2 or meeting a node that has higher utility than 0.8. If node n_i is asked to forward a message to n_9, since the utility between n_i and n_9 is an indirect utility through n_1, n_i forwards the message to n_1 or a node that has a higher utility than 0.6.

Algorithm 1 shows the pseudocode for duration utility calculation, in which every node n_i in the system periodically checks its connectivity. When node n_j moves into the transmission range of n_i, n_i and n_j exchange their utility tables and update their own utility table accordingly based on Formula (5.9). For example, $U_{(i,j)} = 0.4$ and $U_{(j,7)} = 0.5$. Then, $U_{(i,j)} * U_{(j,7)} = 0.2 > U_{(i,7)} = 0.1$. Therefore, n_i changes the entry of "n_7, 0.1, N/A" in its utility table to "n_7, 0.2, n_j". Also, n_i records the duration of the meeting with n_j. At each update time period T, n_i updates the duration utilities between itself and all other nodes

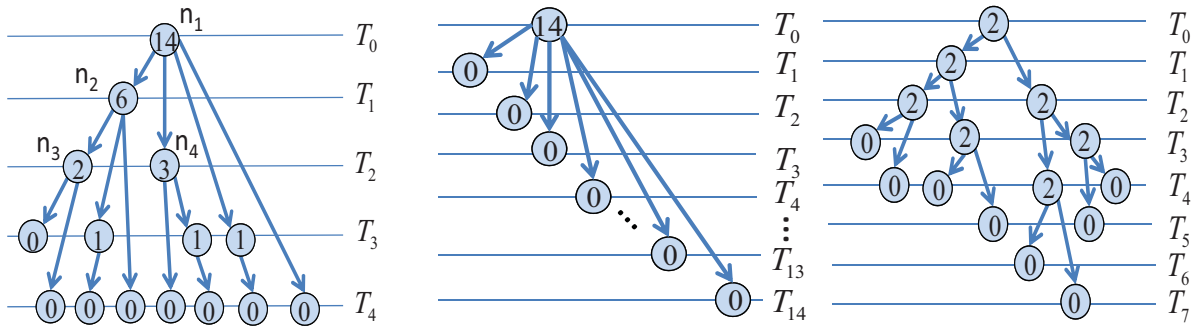
according to Formulas (5.9) and (5.10). Since testing whether a utility for a node exists in a utility table takes constant time by using a hash table and the utility calculation can be finished in constant time, the whole duration utility calculation process can be finished in constant time.

Theorem 5.2.1 *SEDUM has a loop-free route from a source node to a destination node.*

Proof Because of the movement patterns of nodes in the network, the duration utility between each pair of nodes will converge to a stable value that can statistically reflect the communication capacity between the two nodes. Since SEDUM uses unidirectional message routing, a message is always forwarded to a node with higher delivery utility; thus, a routing loop will not occur.

Algorithm 1 Pseudocode for duration utility calculation executed by n_i.

```
1: //When meeting other nodes;
2: if meet node n_j then
3:    Exchange utilities that has been updated since last time meet with n_j
4:    if U_(i,j) exists in n_i's utility table then
5:       for each node n_k in updated utilities do
6:          if U_(i,j) * U_(j,k) > U_(i,k) then
7:             Update U_(i,k) using Formula (5.9)
8:          end if
9:       end for
10:   end if
11:   Record the contacting time with n_j
12: end if
13: //Periodically update its utilities;
14: if currentTime=updateTime then
15:   updateTime+=T
16:   for each meeting node n_j in the last time period T do
17:      Calculate U_(i,j) using Formula (5.9)
18:      if U_(i,j)_old exists in its utility table then
19:         Update U_(i,j) using Formula (5.10)
20:      end if
21:   end for
22: end if
```



(a) Optimal tree replication (b) Source tree replication (c) Binary tree replication

Figure 5.4: Message replication algorithms.

5.2.3 Multi-Copy Routing

SEDUM uses a multi-copy routing method to increase the probability that a message is successfully delivered to a destination node. Since too many replicas will result in high node resource consumption, SEDUM aims to minimize the number of replicas of a message while achieving the desired routing delay. SEDUM can arrive at this minimum number based on network size, meeting interval, and desired transmission delay. We will introduce the details of this calculation in Section 5.3.2.

There are two requirements for the replication algorithm: (1) a message should be replicated quickly and (2) the algorithm can terminate the replication process after exactly N_c replicas (including the source message) are generated. To meet the requirements, SEDUM adopts the optimal tree replication algorithm [86]. In this algorithm, if node n_i is responsible for creating x replicas, when it meets node n_j, n_i sends a copy to n_j. Also, it entitles n_j to be responsible for half of its remaining responsibility; that is, it entitles n_j to replicate $\lfloor \frac{x-1}{2} \rfloor$ copies, and itself is responsible for the other $\lceil \frac{x-1}{2} \rceil$ replicas. Each replica node conducts the same operation until every node has no more responsibility.

Figure 5.4(a) shows an example of the optimal tree replication algorithm. The number in a circle represents the number of replicas a node should create. We use epoch to denote the

time step (T_i) in which a replica node replicates a message to a non-replica node. Assume SEDUM allows each message to have $N_c = 15$ copies for message routing. Then, source node n_1 needs to create an additional $x = N_c - 1 = 14$ replicas in the network. It entitles the first meeting node n_2 to create $\lfloor \frac{(N_c-1)-1}{2} \rfloor$=6 replicas at the first epoch T_1 and keeps the responsibility to create the remaining $\lceil \frac{(N_c-1)-1}{2} \rceil$=7 replicas to itself. At epoch T_2, n_1 entitles the second meeting node n_4 to create $\lfloor \frac{\lceil \frac{(N_c-1)-1}{2} \rceil - 1}{2} \rfloor = \lfloor \frac{6}{2} \rfloor$=3 replicas. At the same epoch, n_2 entitles its meeting node n_3 to create $\lfloor \frac{\lfloor \frac{(N_c-1)-1}{2} \rfloor - 1}{2} \rfloor$=2 replicas, and itself is responsible for the remaining $\lceil \frac{\lfloor \frac{(N_c-1)-1}{2} \rfloor - 1}{2} \rceil$ replicas. Then, in epoch T_3, nodes n_1, n_2, n_3, and n_4 entitle their next meeting nodes with half of their own replication responsibility. The process repeats until each node completes its replicating task. The algorithm needs only $\Theta(\log_2 N_c)$ time steps or 4 steps to replicate 14 copies in this example.

The optimal tree replication algorithm performs better than the source tree replication algorithm and the binary tree replication algorithm [86]. In the source tree replication algorithm, only a source node can replicate a message to others. As shown in Figure 5.4(b), the source node initially tries to create 14 replicas in the network. Since a replica is created only when the source node meets a new node, it takes $\Theta(N_c)$ epochs to create N_c replicas, or 14 epochs to replicate 14 copies in this example. Figure 5.4(c) shows an example of a binary routing tree algorithm where each node can only replicate a message to two other nodes. It requires $\Theta(\log_2 N_c)$ epochs to replicate N_c copies or 6 steps for 14 copies in this example.

Although both the binary tree replication algorithm and optimal tree replication algorithm have $\Theta(\log_2 N_c)$ replication epochs, the former's replication process is slower than the latter on average. The reason is that in the optimal tree replication algorithm, the nodes entitled to replicate messages keep replicating messages until each node has no additional message needing replication. At this time, N_c replicas are generated in the system. In contrast, in the binary tree replication algorithm, the nodes that are entitled to replicate messages stop replicating messages after generating two replicas. Even if the nodes meet

other nodes that do not have any replicas before the entire replication processes complete, they cannot replicate messages. For example, in Figure 5.4(a) at epoch 3, 8 nodes are able to replicate the message to others in the optimal tree replication algorithm, while only four nodes can replicate messages to others in the source tree replication algorithm as shown in Figure 5.4(b). Also, as shown Figure 5.4(c), the binary tree replication algorithm cannot terminate the replication process after N_c replicas are generated in the system.

5.2.4 Buffer Management

Because of the intermittent connections between nodes in DTNs, each node uses a buffer to store the messages for transmission. When two nodes meet each other, since the communication time between two nodes is limited, the order of message transmission affects the transmission throughput and delay of a DTN. Since the size of a buffer is limited, whether or not to accept an incoming message and which message is selected to drop when the queue is full also affects the delivery throughput and delay of a DTN. Additionally, SEDUM routes multiple copies of a message in the network. If one replica is successfully delivered, deletion of other replicas of the message in time to leave space for undelivered messages is also important. Therefore, we must have an effective buffer management mechanism to address these problems to increase the network throughput and reduce transmission delay.

Figure 5.5 shows the structure of a message header. The total size of the message header is 24 bytes. *Source*, 4 bytes, indicates the ID of the source node that generates the message. *Sequence number*, 4 bytes, indicates the sequence of the messages generated from the same source node. *Destination*, 4 bytes, indicates the ID of the destination node that should receive the message. *Timestamp*, 8 bytes, records a message's creation time, and *priority*, 4 bytes, indicates the priority of a message determined by the tolerable delay specified by the source node. In a DTN application, different messages may have different tolerable delays. For example, voice messages should have lower tolerable delays than text

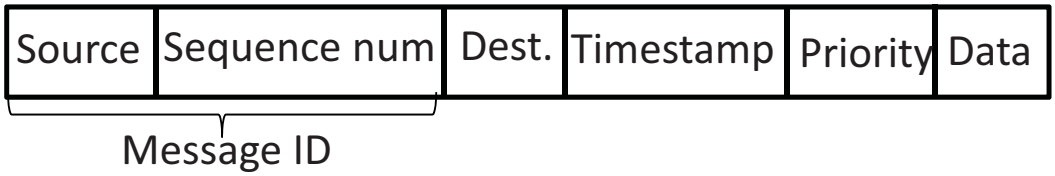


Figure 5.5: The structure of the message head.

messages. SEDUM allows a source node to specify the tolerable delay of its message to ensure timely message delivery. SEDUM includes specification for priority levels and assigns higher priority levels to messages with lower delay tolerance.

As shown in Figure 5.6, each node orders the messages in its buffer according to the messages' priorities (priority 1 has a higher priority than priority 2) and timestamps. The messages are ordered in descending order of their priorities. In each priority level, the messages are sorted in ascending order of their timestamps. A larger timestamp means a shorter lifetime since a message was initiated. For example, in the group of priority 1 messages, M_{11} was created earlier than M_{73}, so M_{11} has higher priority than M_{73}. When two nodes, say n_i and n_j, meet each other, the messages at n_i whose destination corresponds to n_j are transmitted first. For other messages, n_i fetches a message from its buffer in a top-down manner. n_i compares its utility and n_j's utility to the message's destination. Recall that we use U_i to denote node n_i's utility to a message's destination. For each fetched message, if $U_i < U_j$, n_i forwards the message to n_j. Node n_j conducts the same operations. For example, in Figure 5.6, M_{73}, M_{28}, M_{91} and M_{32} satisfy $U_1 < U_2$, then n_1 sends these messages to n_2 in sequence. Therefore, the messages with higher priority are sent out first. Within each priority level, the messages with longer transmission delay are sent out first. The consideration of priority helps to deliver messages within their specified tolerant delay. The consideration of timestamps ensures that the longer a message remains in the network, the higher the chance that it is delivered out of the buffer first, which avoids having a message always stuck in a buffer and decreases the message delivery delay of the DTN.

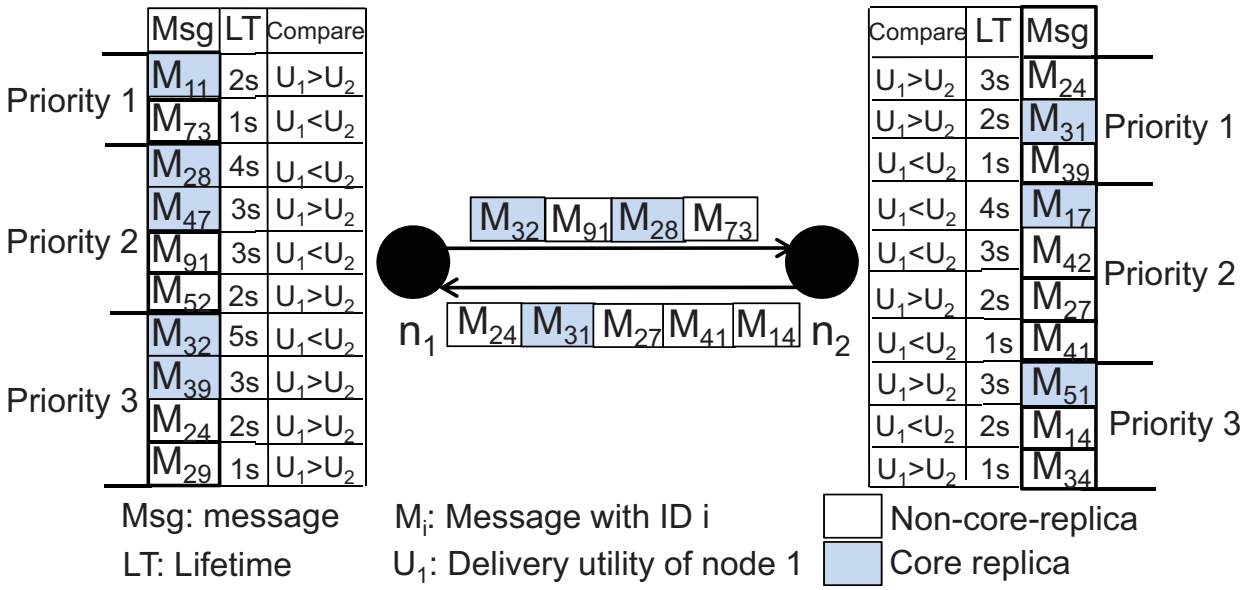


Figure 5.6: An example of message exchange.

Next, we discuss how a node addresses an incoming message. To avoid losing messages due to buffer congestion, we adopt the congestion control method in [12]. That is, for a number of message copies, a source node initially decides a core-replica, which is the replica stored at the neighbor with the highest delivery utility to the destination node. A core-replica in a buffer cannot be replaced, but it can replace a non-core-replica in a buffer if the buffer is congested. In SEDUM, a node can be selected by a number of nodes as a relay node, requiring it to store a number of messages with different delivery utilities. To make wise use of the limited buffer resources, messages with higher utilities should have a higher priority to use the buffer. Thus, more messages can be delivered to their destinations during a certain time period. Core-replicas have a higher priority to stay in the buffer than non-core-replicas. When a node with a full buffer receives a message, if the message is a core-replica, it replaces the non-core-replica with the smallest utility in the buffer. If the incoming message is a non-core-replica with utility U_j, the node finds non-core-replicas in its buffer whose utility is lower than U_j, then replaces the non-core-replica with the lowest

utility. If all non-core-replicas's utilities are larger than U_j, the node drops the incoming message. SEDUM's buffer management method ensures that the core-replica of a message must remain in the system, thus guaranteeing successful delivery of each message. Also, it gives higher buffer priority to higher-utility messages, thus enhancing system throughput.

In SEDUM, when nodes n_i and n_j meet each other, they transmit messages according to Algorithm 2. Specifically, they exchange their utility tables. According to n_j's utility table, n_i finds in its buffer those messages that could have higher utilities if residing in n_j and forwards those messages to n_j. If n_j has free space in its buffer, it accepts the incoming messages. If n_j's buffer is full of core-replicas, it rejects n_i's messages. If the incoming message M_i is a non-core-replica and there is no message in the buffer that has a lower utility than M_i's, n_j also rejects M_i. Otherwise, the incoming message M_i replaces the message with the lowest utility in the buffer. We have implemented the buffer as a min-heap [26]. Therefore, for each message, we obtain the message with minimum utility in constant time. As the heap structure is maintained with time complexity O(logn) for each message, the complexity of the whole algorithm is O(logn) for each message.

Delivered Message Deletion. We define a message's ID as the concatenation of its source ID and sequence number. The replicas, especially the core-replicas, of the delivered messages must be deleted in a reasonable time to free buffer space for undelivered messages. In SEDUM, every node keeps a delivered message list (*deliveredMsgList*) that records the IDs of all delivered messages. When node n_i meets node n_j, they exchange their (*deliveredMsgList*). Each node then deletes the messages in its buffer indicated in the other's (*deliveredMsgList*) and merges this list with its own (*deliveredMsgList*). The node that does not update its (*deliveredMsgList*) will continuously hold the delivered message copy until it meets the destination node or the messages are replaced by other messages according to buffer management algorithm. After a message is delivered, the last hop node puts the delivered message ID into the (*deliveredMsgList*), which will be exchanged among nodes in the

Algorithm 2 Pseudocode for message transmission from n_i to n_j.

```
1: //Sending messages;
2: Send messages with Dest.=n_j to n_j
3: for each message M in the buffer do
4:    if U_j > U_i then
5:       Send message M to n_j
6:    end if
7: end for
8: //Receiving messages;
9: for each received message M with U_j do
10:   if its buffer is not full then
11:      Accept message M
12:   else if all messages in buffer are core-replicas then
13:      Reject message M
14:   else if message M is a non-core-replica then
15:      if no message has utility lower than U_j then
16:         Reject message M
17:      else
18:         replace the messages with the lowest utility with message M
19:      end if
20:   else
21:      replace the messages with the lowest utility with message M
22:   end if
23: end for
```

system. Even if a node misses an update to the (*deliveredMsgList*), it will finally delete the messages in a later meeting with other nodes with high probability because of the flooding feature of the notification message. Even if it is not deleted in time and is sent to the destination again, the destination will discard the message that it has already received. To restrict the size of the (*deliveredMsgList*), each node periodically deletes outdated message IDs in (*deliveredMsgList*). To guarantee that one of the replicas of a message is delivered in the system before the message's ID is discarded from all (*deliveredMsgList*), the ID discarding period should be set as the upper bound of message transmission time in the system within which a message should be delivered.

5.3 Performance Analysis

5.3.1 Analysis of the Routing Protocol

The single-copy routing protocols generate lower overhead but lead to a longer delivery delay than the multi-copy routing protocols. Epidemic routing can produce short delay in a lightly loaded transmission environment since a message is delivered to the destination along the shortest path by flooding. However, flooding consumes significant energy resources, which are precious to mobile-devices. Small and Haas [86] indicated that restriction of transmission traffic can save energy in the network. Multi-copy routing can reach a tradeoff between the single-copy routing and epidemic routing. Therefore, SEDUM creates N_c ($N_c \ll N$) replicas for a message to increase the probability of a message being delivered to its destination node (i.e., offloading probability) while reducing resource consumption in the network.

Theorem 5.3.1 *If the number of replicas per message in multi-copy routing is large, adding more replicas leads to a linear increase of energy consumption but a negligible delivery delay*

decrease.

Proof Replicating $N_c - 1$ copies of a message consumes $(N_c - 1)E$ amount of energy, where E is the average energy consumption for each transmission. Suppose the average message offloading probability of each node is p. If N_c is constant over the entire lifetime of the message, the offloading delay follows a geometric distribution with mean $\frac{1}{N_c p}$. If we create one more replica, the message offloading delay is reduced by $\frac{1}{N_c p} - \frac{1}{(N_c+1)p} = \frac{1}{(N_c+1)N_c p}$. Therefore, if N_c is large, as the number of replicas of the message increases, the rate of message offloading delay decreases non-linearly while the energy consumption increases linearly.

As mentioned earlier, SEDUM routing consists of three phases: replicating, forwarding, and clearing. Since each of the N_c relay nodes for a message looks for a routing path independently in the forwarding phase, delay in the replicating phase adversely affects the delivery delay of the message in the forwarding phase and sequentially deteriorates the whole system transmission efficiency.

Theorem 5.3.2 *Suppose the average total replication delay of a message is T_r. Then the average delivery delay is in the order of $O(T_r)$.*

Proof The meeting time of two randomly selected nodes is exponentially distributed with average T_m [13]. The expected duration of the forwarding phase is $T_f = \frac{T_m}{n_c}$, where n_c is the number of generated replicas when a replica meets the destination. The average number of replicas at time t is $\frac{N_c}{T_r}t$ $(t \in [1, T_r])$. Then, the average forwarding delay is:

$$T_f = \frac{\sum_{t=1}^{T_r} \frac{T_m}{(N_c/T_r)\cdot t}}{T_r} = \sum_{t=1}^{T_r} \frac{T_m}{N_c \cdot t}. \tag{5.11}$$

The average delivery delay of a message is therefore:

$$\sum_{t=1}^{T_r} (\frac{t}{T_r} + \frac{T_m}{N_c \cdot t}) = \frac{T_r + 1}{2} + \frac{T_m}{N_c} O(lnT_r) = O(T_r). \tag{5.12}$$

Therefore, to reduce the delivery delay, we must reduce the replication delay. Thus, in the replicating phase of SEDUM, a source node should replicate a message to its neighbors quickly regardless of their utilities, instead of only replicating the message to high-utility nodes as in the forwarding phase. The initial low-utility nodes will meet high-utility nodes later based on the routing algorithm. Thus, reducing the replication time can reduce the delivery delay.

In Section 5.2.3, we explained the main three message replication algorithms: source tree, binary tree, and optimal tree. We now compare the delay performance of the three algorithms.

Theorem 5.3.3 *The optimal tree replication algorithm can reduce the delay of the source tree replication algorithm by:*

$$\sum_{i=1}^{N_c}(\frac{N-1}{N-i}) - \sum_{i=1}^{\lfloor log_2 N_c \rfloor}(\frac{N-1}{N-2^i+1}), \tag{5.13}$$

where N is the number of nodes in the system and N_c is the number of replicas in the system.

Proof In the source tree replication algorithm, message replication occurs only when a source node meets a non-replica node. Assume it has generated l replicas. The probability that it sends the $(l+1)^{th}$ replica to a new node follows a geometric distribution with mean $\frac{N-l}{N-1}$. Therefore, the average delay for replication is $\frac{N-1}{N-l}$. Since the number of epochs to replicate l replicas is $l-1$, if N_c replicas are to be created in the system, the average delay of the full creation is $\sum_{i=1}^{N_c}(\frac{N-1}{N-i})$. During the i^{th} epoch in the optimal replication algorithm, the probability of creating a replica node equals $\frac{N-2^i+1}{N-1}$. Since the number of epochs to replicate N_c replicas is $\lfloor log_2 N_c \rfloor$, the average delay for creating N_c replicas is $\sum_{i=1}^{\lfloor log_2 N_c \rfloor}(\frac{N-1}{N-2^i+1})$.

Theorem 5.3.4 *To replicate N_c replicas, the binary tree replication algorithm takes $log_2 N_c + 2$ epochs and the optimal tree replication algorithm takes $log_2 N_c$ epochs.*

Proof In the optimal tree replication algorithm, since the nodes entitled to replicate messages continue replicates until N_c replicas are generated in the system, the algorithm requires $log_2 N_c$ epochs to complete. In the binary tree replication algorithm, it takes $log_2 N_c$ epochs to entitle nodes to replicate messages. After that, it takes two additional epochs for the last entitled node to replicate messages. The total number of replication epochs equals $log_2 N_c$+2.

Theorem 5.3.5 *The average delivery delay $\overline{T}_d$ for the SEDUM routing protocol follows $O(\sqrt{N}) > \overline{T}_d > O(\log N)$, where N is the number of nodes in the system.*

Proof In the worst situation of the forwarding phase in SEDUM, where replica nodes cannot find a higher utility node for relaying, SEDUM becomes a two-hop multi-copy routing protocol, the delay of which is $O(\sqrt{N})$ [65]. Conversely, if replica nodes in SEDUM can always find a relay node with higher utility, SEDUM becomes an optimal redundancy multi-hop routing protocol [65], the delay of which is $O(\log N)$.

5.3.2 Analysis of the Message Replication Process

In multi-copy routing, too many replicas of a message generate high overhead while too few replicas of a message may lead to a long delivery delay of the message. The number of replicas of a message is an important issue that affects routing performance. To find the smallest number of message replicas that can guarantee a specified routing delay, we build a Markov chain to analyze the message replication process. Figure 5.7 shows a Markov chain that models a message replication process. In the figure, $\pi(l)$ ($l \in [1, N_c]$) denotes the network state where l replicas (including the original message) have been generated before any of the replicas meet the destination. END denotes the state that a replica meets the destination and the message transmission completes. The arrows between states in the figure indicate the state changing direction. We use $p\big(\pi(l), \pi(l+1)\big)$ to denote the probability that

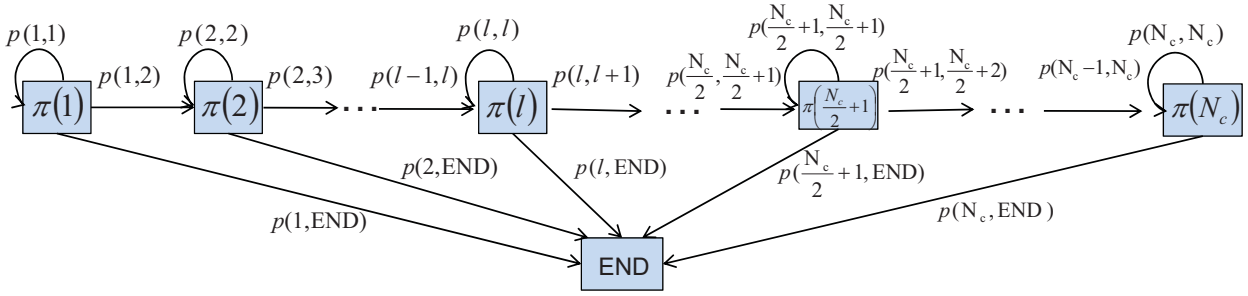


Figure 5.7: A Markov chain that models a message replication process.

a non-replica node in network state $\pi(l)$ receives a replica, which changes the network state to $\pi(l+1)$.

We call the replica nodes that are entitled to replicate messages to other non-replica nodes *entitled nodes*, and the replica nodes without replication responsibility *non-entitled nodes*. In the optimal tree replication algorithm, when $l < \frac{N_c}{2}$, every replica node is an entitled node because every replica node is entitled to replicate at least one replica. When $l > \frac{N_c}{2}$, only a portion of the replica nodes are entitled nodes. In the case of $l \in [1, \frac{N_c}{2}]$, when two nodes meet each other, because there exist l replica nodes, the probability that one node is an entitled node equals $\frac{l}{N}$ and the probability that the other node is a non-replica node is $\frac{N-l-1}{N-1}$. The "1" in $(N-l-1)$ indicates the destination, and the "1" in $(N-1)$ indicates the first meeting node. Also, the probability of node n_i meeting node n_j is the same as the probability of n_j meeting n_i. Then, $p\big(\pi(l), \pi(l+1)\big) = 2 \cdot \frac{l}{N} \cdot \frac{N-l-1}{N-1}$.

We consider the case when $l \in [\frac{N_c}{2}+1, N_c]$ (i.e., in the last epoch T_l) first to calculate the number of entitled nodes. Suppose there are x entitled nodes and $l-x$ non-entitled nodes in epoch T_l. The $l-x$ non-entitled nodes in epoch T_l are separated from the $(1-x)/2$ entitled nodes in epoch T_{l-1}. Since the total number of replica nodes in epoch T_{l-1} is $N_c/2$, $x + (l-x)/2 = N_c/2$. Then, in epoch T_l, the number of entitled nodes is $x = N_c - l$ and the number of non-entitled nodes is $l - x = 2l - N_c$. Therefore, the probability that an entitled node meets a non-replica node is $2 \cdot \frac{N_c-l}{N} \cdot \frac{N-l-1}{N-1}$. That is, $p\big(\pi(l), \pi(l+1)\big) = 2 \cdot \frac{N_c-l}{N} \cdot \frac{N-l-1}{N-1}$.

If one of the replica nodes meets the destination node in the next epoch, the message transmission finishes. Therefore, the probability that the message can be delivered in state l in the next epoch is $P(l, \text{END}) = 2 \cdot \frac{l}{N} \cdot \frac{1}{N-1}$, where $\frac{l}{N}$ is the probability that one node is a replica node and $\frac{1}{N-1}$ is the probability that the other node is the destination when two nodes meet. In all other encountering cases, the network state stays the same. That is, $p\big(\pi(l), \pi(l)\big) = 1 - p\big(\pi(l), \pi(l+1)\big) - p\big(\pi(l), \text{END}\big)$. In conclusion, the transition probability in the Markov chain between two states is:

$$\begin{cases} p\big(\pi(l), \pi(l+1)\big) = 2 \cdot \dfrac{l}{N} \cdot \dfrac{N-l-1}{N-1}, l \in [1, \dfrac{N_c}{2}]; \\ p\big(\pi(l), \pi(l+1)\big) = 2 \cdot \dfrac{N_c - l}{N} \cdot \dfrac{N-l-1}{N-1}, l \in [\dfrac{N_c}{2} + 1, N_c]; \\ p\big(\pi(l), \text{END}\big) = 2 \cdot \dfrac{l}{N} \cdot \dfrac{1}{N-1}; \\ p\big(\pi(l), \pi(l)\big) = 1 - p\big(\pi(l), \pi(l+1)\big) - p\big(\pi(l), \text{END}\big). \end{cases} \tag{5.14}$$

Suppose the average meeting interval between two nodes is T_m. We use $T_d(l)$ to denote the latency to reach network state $\pi(l)$, i.e., to replicate l replicas:

$$T_d(l) = p\big(\pi(l-1), \pi(l)\big) \cdot \big(T_d(l-1) + T_m\big) + p\big(\pi(l), \pi(l)\big) \cdot \big(T_d(l) + T_m\big). \tag{5.15}$$

When the number of replicas that a message is allowed to generate equals N_c, the average message delivery delay $T_d(END)$ is:

$$T_d(\text{END}) = \sum_{i=1}^{N_c} \big((T_d(i) + T_m) \cdot p(i, \text{END})\big). \tag{5.16}$$

According to Formulas (5.14), (5.15), and (5.16), given an average node meeting interval T_m, the number of nodes in the network N, and the number of replicas of a message N_c, the average delivery delay $T_d(\text{END})$ can be calculated. For example, when N=10, N_c=4 and T_m=1s, we can construct a Markov chain as shown in Figure 5.8 based on Formula (5.14).

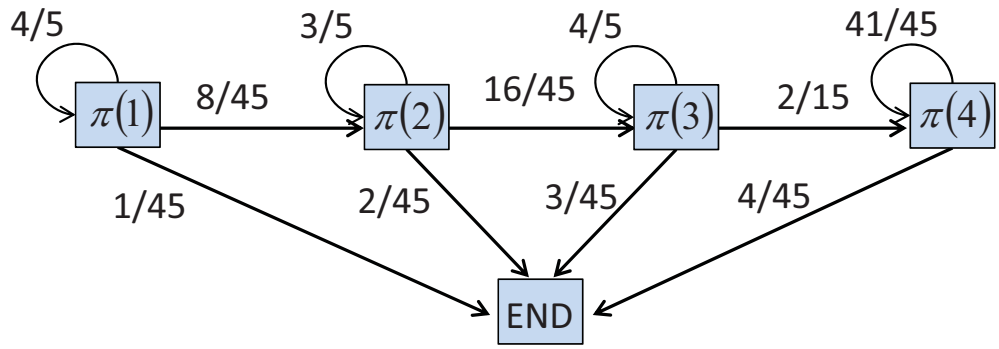


Figure 5.8: An example of a constructed Markov chain.

Then, based on Formula (5.15), we obtain:

$$\begin{cases} T_d(1) = \frac{4}{5}(T_d(1)+1) \\ T_d(2) = \frac{4}{9}T_d(1) + \frac{35}{18} \\ T_d(3) = \frac{16}{9}T_d(2) + \frac{52}{9} \\ T_d(4) = \frac{3}{2}T_d(3) + \frac{47}{4}. \end{cases} \tag{5.17}$$

Then, based on Formula (5.16), we obtain:

$$T_d(END) = \frac{1}{45}T_d(1) + \frac{2}{45}T_d(2) + \frac{3}{45}T_d(3) + \frac{4}{45}T_d(4). \tag{5.18}$$

By solving Equation (5.18), we retrieve $T_d(END) = \frac{34}{9}s$. Thus, given a delay tolerance $\mathcal{T}$, we can adaptively adjust the value N_c to guarantee $T_d(\text{END}) \leq \mathcal{T}$.

5.4 Summary

In this chapter, we propose a duration utility that fully captures both stranger familiar and collocation attributes. We theoretically prove that a duration utility can more accurately reflect node communication capacities. We then propose the SEDUM routing protocol for

DTNs that fully exploits node movement patterns in the social network to increase delivery throughput and decrease delivery delay while generating low overhead. SEDUM replicates a new message to a certain number of nodes, which then hold the replicas until meeting other nodes with higher duration utilities to the destinations. A message is forwarded in this way until one of its replicas reaches its destination. SEDUM includes a buffer management mechanism to improve performance. We also introduce a method using a Markov chain to calculate the minimum number of copies of a message to achieve a given delivery delay.

In the next Chapter, we will present the evaluation results of the integrated system in Chapter 3, ARM in Chapter 4, and SEUMN in this Chapter.

Chapter 6

Performance Evaluation

In this chapter, we validate the theoretical results that are presented in Chapter 3 and present the simulation results of the hierarchical account-aided reputation management system proposed in Chapter 4. We also show the simulation results of the social network and utility based distributed multi-copy routing protocol proposed in Chapter 5.

The rest of this chapter is structured as follows. In Section 6.1, we evaluate the trustworthiness of the proposed integrated system. In Section 6.2, we show the simulation results of the hierarchical account-aided reputation management system. In Section 6.3, we evaluate the performance of the social network and utility based distributed multi-copy routing protocol. We finally summarize our results in Section 6.4.

6.1 Evaluation of the Integrated Reputation/Price System

6.1.1 Comparison of Incentives of Different Systems

In this section, we evaluate the effectiveness of the incentives in the defenseless system, reputation system, price system, and integrated system in a repeated game, where the nodes can change their interaction strategies adaptively. We developed a simulator based on the Monte Carlo method [20], in which two nodes are repeatedly and randomly paired up for interaction until the reputation values of nodes are converged. At every game round, each randomly formed pair of nodes have an interaction. That is, the nodes send a packet to each other and drop or forward their received packet from the other. In the simulation, 100 nodes are i.i.d. in the system. 50 nodes are cooperative and 50 nodes are non-cooperative at the start. The number of players using a strategy in the next round was set to the product of the *relative success rate* of this strategy in the previous game round and the node population.

In the simulation, the packet forwarding reward is $m_r = 2$ units, the packet forwarding price is $m_p = 1$ units, the transmission benefit is $p = 4$ units, and the transmission cost is $c = 2$ units. The initial reputation value for each node is 1.0 and the reputation threshold is $T_R = 0.3$. The maximum reputation value is 1.0. Every time a node helps forward a packet, its reputation value is increased by 0.1. Otherwise, its reputation value is reduced by 0.1. We define the density of the (non-)cooperative nodes as the percent of the nodes employing the (non-)cooperative strategy among all nodes. In each figure, the analytical results calculated by Equation (3.30) are included based on the simulation parameters with the individual payoff matrix.

Figure 6.1 shows the density change of cooperative and non-cooperative nodes in a defenseless MANET. The figure shows that after several interactions, the selfish nodes dominate the population of the system, because in the defenseless system, the non-cooperative strategy is the NE, although not Pareto-optimal. Therefore, the nodes using a non-cooperative strategy can receive much more payoff than the nodes using a cooperative strategy. Since the number of nodes using a strategy depends on the relative success rate of the nodes using

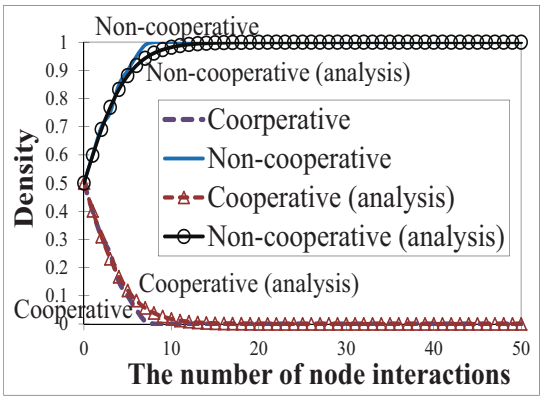


Figure 6.1: The defenseless system.

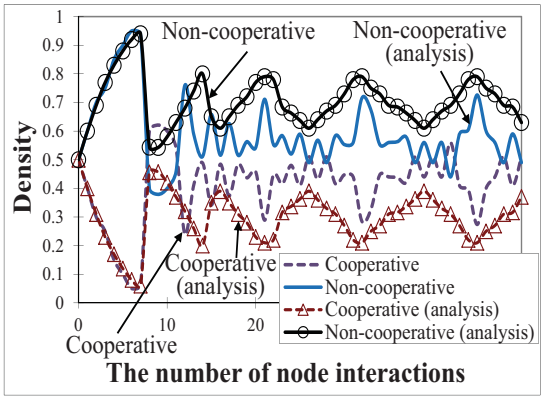


Figure 6.2: The reputation system.

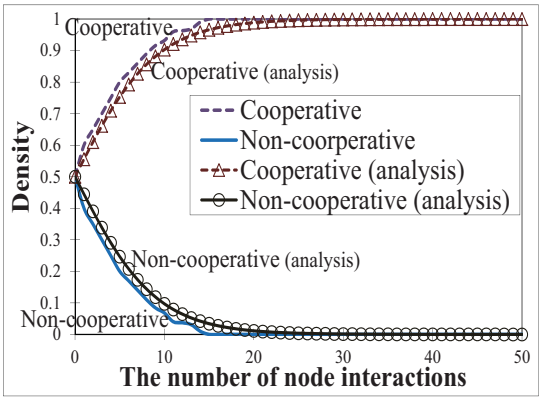


Figure 6.3: The price system.

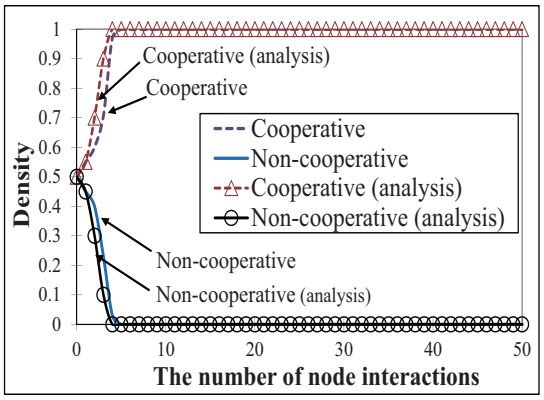


Figure 6.4: The integrated system.

this strategy in the last round, the number of players using a cooperative strategy decreases sharply. Therefore, the defenseless MANET without any cooperation incentive or misbehavior detection mechanism will finally collapse. Also, from the figure we can see that the simulation results are consistent with the analytical results in Proposition 3.5.2.

Figure 6.2 shows the density change of cooperative and non-cooperative nodes in a MANET with a reputation system. The figure indicates that in the first 8-9 interactions, the density of non-cooperative nodes increases and the density of cooperative nodes decreases, because during these game rounds, (I_i, I_j) is the NE continually. The non-cooperative strategy can bring much more payoff than the cooperative behavior, which results in a dramatic

population decrease of cooperative nodes. However, when the reputation values of some nodes fall below the reputation threshold, the payoffs of (I_i, I_j), (C_i, I_j), and (I_i, C_j) turn to (0, 0), according to Table 3.4. Therefore, the cooperative strategy is the NE and Pareto-optimal. At this time, since the cooperative action can generate much higher payoff than the non-cooperative action, the population of cooperative nodes increases. However, after the reputation values of the nodes increase above the threshold, they will again choose (I_i, I_j). Then, the density of selfish nodes increases. The figure also shows that the percentages of cooperative nodes and selfish nodes finally approach a constant value, which is the reputation threshold value. This result closely matches Proposition 3.3.2, which indicates that the strongest incentive provided by a reputation system will result in a situation where nodes maintain their reputation close to and above the reputation threshold. The simulation results agree with our analytical result for Proposition 3.5.2.

Figure 6.3 shows the density change for cooperative and non-cooperative nodes in a MANET with a price system. The figure shows that cooperative nodes eventually dominate the population in the system because nodes are rewarded for providing packet forwarding services to others and charged for receiving packet forwarding service from others. The system increases the payoff of the cooperation strategy and decreases the payoff of the non-cooperation strategy. Therefore, (C_i, C_j) is the NE, and the density for the cooperative nodes increases sharply and that of the selfish nodes decreases rapidly. These results agree with Proposition 3.4.1, Proposition 3.4.2, and Proposition 3.5.2.

Figure 6.4 shows the density change of cooperative and non-cooperative nodes in a MANET with an integrated system. The integrated system can distinguish the service quality of nodes based on their reputation values, which reflects their cooperation levels. In the integrated system, a lower-reputed node receives a lower payoff, while a higher-reputed node receives a higher payoff for providing service. Because the cooperation strategy becomes both the NE and Pareto-optimal, a cooperative node earns a much higher payoff than a

non-cooperative node. Therefore, the number of cooperative nodes is more than the number of selfish nodes. Meanwhile, as the number of game rounds increases, the reputations of the nodes similarly increase. Consequently, the payoff for the (C_i, C_j) also increases and the number of selfish nodes in the integrated system drops much faster than the price system. Therefore, the integrated system can provide higher incentives than other systems to encourage the cooperation of the nodes. The simulation results are consistent with our analytical result in Proposition 3.5.2.

6.1.2 Evaluation of the Reputation System

Since the Monte Carlo method cannot simulate a network scenario, we further investigate the effectiveness of these systems for selfish node detection in a MANET scenario with NS-2 [4]. In the simulated MANET, 100 nodes are i.i.d. in a 500m $\times$ 500m square area, with a transmission range of each node at 250m. Each node randomly selects a position in the area and moves to the position at a speed randomly selected within $[10 - 20]$m/s. In the test, we first assign each node a reputation value randomly chosen from $[0, 1]$. We then randomly select 10 source nodes in every second. Each of the 10 nodes sends a packet to a randomly chosen neighbor. If the neighbor's reputation value is lower than T_R, it drops the packet and its reputation value is decreased by 0.1. Otherwise, the neighbor forwards the packet and subsequently its reputation value is increased by 0.1. The simulation time for each test is 10,000s.

Figure 6.5 shows the initial reputation values of all nodes in the system. The reputation values are spread over the range $[0, 1]$. Since the nodes are punished only when their reputation values fall below T_R, they can randomly drop packets to save energy when their reputation values are above the threshold. When their reputation values are below T_R, they cooperate in packet forwarding to increase their reputation values above T_R to avoid being punished. We test the reputation values of nodes in the system when T_R equal to 0.3 and

0.7, respectively. Figure 6.6 shows the final reputation values of all nodes in the system after 10,000s. We can see that the reputation values of all nodes converge to the reputation threshold in each case, a result that is consistent with Figure 6.2 and Proposition 3.3.2. By keeping its reputation value just above the threshold, a node can be uncooperative while still avoiding punishment. Therefore, the reputation system cannot provide highly effective incentives to encourage the nodes to be cooperative.

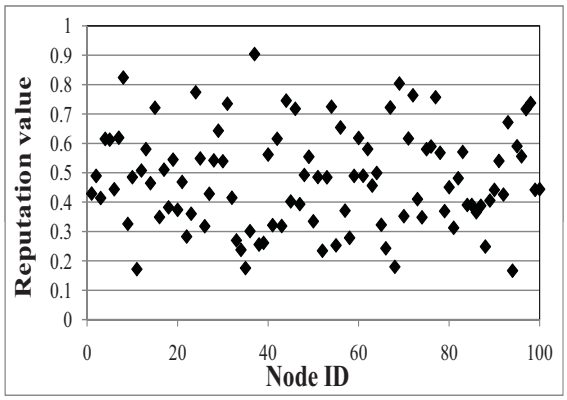


Figure 6.5: Distribution of initial reputation values.

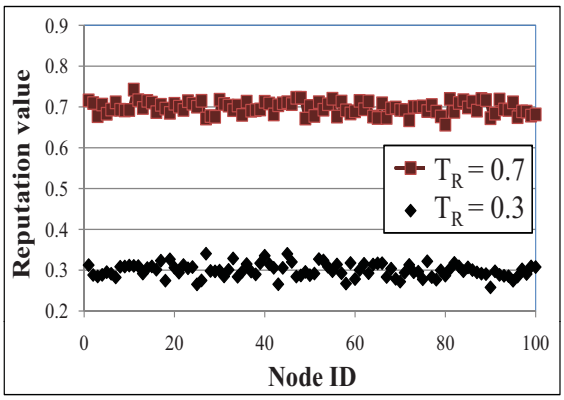


Figure 6.6: Converged reputation values.

We use the *decrease/increase rate* (DIR) to denote the ratio of the reputation decrease rate to the reputation increase rate. The packet drop rate is the total number of dropped packets divided by the total number of received packets. In this experiment, we vary DIR from 1 to 8 with 1 increase in each step, and test the packet drop rate for each DIR in a 10,000s simulation. Specifically, the reputation increase rate is 0.1, and the reputation decrease rate ranges from 0.1 to 0.8 with a 0.1 increase in each step. Figure 6.7 shows the experimental and theoretical results of the packet drop rate versus DIR. The theoretical results are calculated according to Formula 3.10 in Proposition 3.3.3. The figure shows as DIR increases, the packet drop rate decreases. Higher DIR means a node's reputation value decrease for its uncooperative behavior is more than the reputation increase for its cooperative behavior. Thus, with a higher DIR, a node must be cooperative for DIR to

decrease its reputation value due to one-time uncooperative behavior. Since a higher DIR stimulates nodes to be cooperative, the packet drop rate decreases as DIR increases. The measurement results are approximately in line with the theoretical results, with the error bar within 0.05.

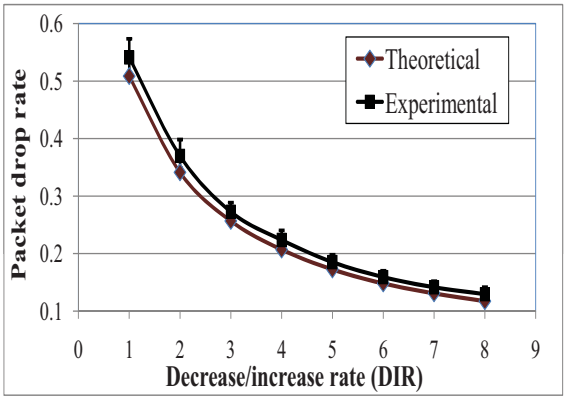


Figure 6.7: Packet drop rate vs. DIR.

Figure 6.8 further shows the packet drop rate versus the DIR and reputation threshold, which exhibits the same phenomenon as shown in Figure 6.7, detailing the relationship between the packet drop rate and DIR. It is very intriguing to see that the reputation threshold does not affect packet drop rate and that the rate is only affected by DIR. As shown in Figure 6.6, node reputation values finally converge to the threshold regardless of the threshold value. Some nodes maintain their reputations just above the threshold. If a node drops a packet, its reputation value falls below the threshold and it must be cooperative for DIR interactions to raise its reputation value above the reputation threshold. This phenomenon is why the packet drop rate is only determined by DIR. Higher DIR leads to lower drop rate and vice versa. This result is very intriguing and consistent with Proposition 3.3.3.

In summary, reputation systems cannot effectively encourage nodes to be cooperative in the system, but only to keep their reputation values around the reputation threshold.

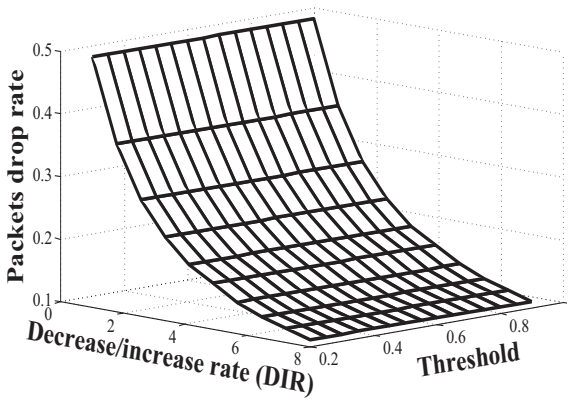


Figure 6.8: Packet drop rate vs. DIR and reputation threshold.

To reduce the packet drop rate, the reputation decrease rate should be higher than the reputation increase rate.

6.1.3 Evaluation of the Price System

In this section, we evaluate how a price system encourages the cooperation of the nodes in the system. The simulation setup and scenario are identical to those at Section 6.1.2, but instead of rating node reputation values, a node pays credits to the forwarding nodes for their services. Since this is a generic price system, we do not consider the details of how nodes pay for the price of packet forwarding service. We assign 1000 credits to each node initially. A packet receiver drops the packet if its account value is above zero. The forwarding price is 50 credits. We use RRP to denote the Ratio of packet forwarding Reward to forwarding Price and test the packet drop rate with different RRPs. Specifically, we initially set the forwarding reward to 25 credits, and then increase it from 50 credits to 350 credits of 50 credit increment in each step. The entire simulation time for each RRP value is 10,000s.

Figure 6.9 shows the experimental and theoretical results of the packet drop rate versus RRP. The theoretical results are calculated based on Equation (3.14) in Proposition 3.4.3.

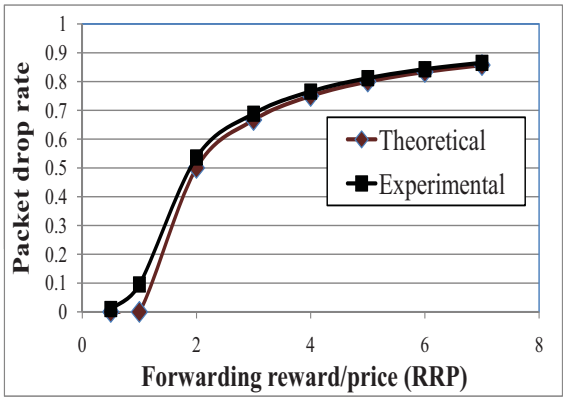


Figure 6.9: Packet drop rate vs. RRP.

The figure demonstrates that the packet drop rate grows as RRP increases, because when the reward is larger than the price, a selfish node can drop more packets and forward fewer packets while still maintaining its account value above zero. Thus, a higher RRP leads to more dropped packets by selfish nodes. These measured results are closely consistent to the theoretical results in Proposition 3.4.3. To restrict the packet drop rate of selfish nodes, the forwarding reward should be less than the forwarding price in a price system.

Packet generating and receiving rates are the number of bits per second a node generates to send out and receives to forward, respectively. We use RGR to denote the Ratio of packet Generating rate and packet Receiving rate of a node. In this experiment, a randomly chosen node i generates and sends packets to m ($m \in [1, 5]$) randomly chosen neighbors at the speed of $2k/s$ for each packet stream. We also randomly choose node i's neighbor j and let it generate and send packets to node i at the speed of $2k/s$. The size of one packet is $2k$. Node i's RGR is varied from 1 to 5 at 1 increment in each step.

Figure 6.10 plots node i's packet drop rate versus its RGR and RRP. The figure shows that as RGR increases, the packet drop rate decreases sharply. Higher RGR means that a node's packet generating rate is faster than its packet receiving rate. That is, the credits needed to pay for the forwarding services are more than can be earned. Insufficient

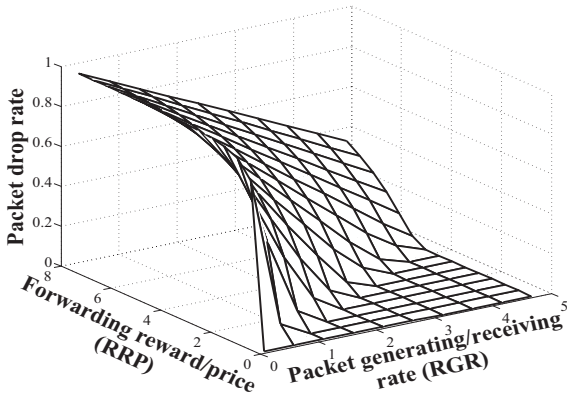


Figure 6.10: Packet drop rate vs. RRP and RGR.

credits stimulate the node to be cooperative. The figure also shows that a larger RGR and smaller RRP make packet drop rates decrease faster. Recall that small RRP and large RGR respectively impose significant effort on reducing the packet drop rate. Under the impact of both factors, the packet drop rate is reduced sharply. Therefore, a node with a high packet generating rate is unlikely to be uncooperative in a MANET using a price system. However, for nodes with a low packet generating rate, they are likely to drop packets since they do not need to earn credits for packet forwarding requests.

6.1.4 Evaluation of the Integrated System

In this section, we demonstrate how an integrated system can improve the effectiveness of detecting selfish nodes and encouraging cooperation in both reputation and price systems. In this experiment, both the reputation increase rate and decrease rate were set to 0.1. The initial reputation value of each node was set to 1 and the reputation threshold was set to 0.2. Each node was initially assigned 1000 credits unless otherwise specified. At every second, ten source nodes are randomly selected, each of which sends a packet to a randomly chosen neighbor. The source node i pays the forwarder 50 credits in the price

system and $50/R_i$ in the integrated system, where R_i is the source node's current reputation. The entire simulation time is 10,000s. In the integrated system, we assume nodes choose the strategy that maximizes their benefit (i.e., the cooperative strategy) with probability $\min(0.8+\frac{\Delta m_p}{m_p}, 1)$, where $\Delta m_p = m_p(t) - m_p(t+1)$. 0.8 and $\frac{\Delta m_p}{m_p}$ are the probabilities that a node is cooperative because of the reputation system and price-based system, respectively.

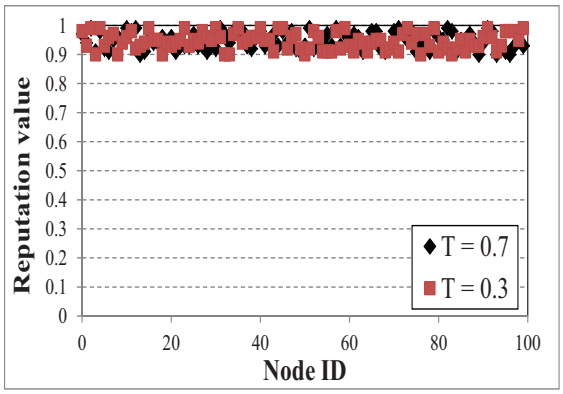


Figure 6.11: Converged reputation value in the integrated system.

Figure 6.11 shows the converged reputation values of nodes in the integrated system after 10000s. As the experiment of Figure 6.6 for a reputation system, we set the reputation threshold of the nodes in the system to $T_R = 0.3$ and $T_R = 0.7$, respectively, and the initial node reputation distribution is shown in Figure 6.5. Comparing Figure 6.11 with Figure 6.6, we see that rather than converging to the reputation thresholds respectively as in the reputation system, the node reputation values in the integrated system are converged to 1. Nodes always choose the action strategy that maximizes their utilities. In the integrated system, the forwarding strategy can provide a node with the best utility. Therefore, nodes always forward packets for others and their reputation values increase to the maximum. In the reputation system, when a node's reputation value is just above the threshold, it does not have incentives to forward others' packets because the forwarding cannot bring about more utility. These results prove the higher effectiveness of the integrated system in cooperation

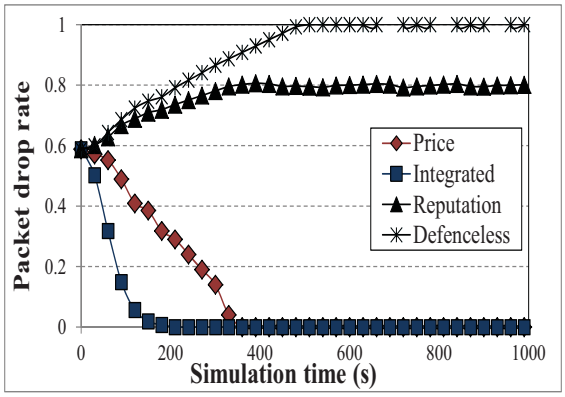


Figure 6.12: Packet drop rate over time.

encouragement than the reputation system.

Figure 6.12 shows the packet drop rates in different systems over the simulation time when the reputation threshold equals 0.2. We see that as time goes on, the packet drop rates of the price-based and integrate systems decrease and those of the reputation and defenseless systems increase. Such differences is because the forwarding strategy can always ensure that the nodes in both the price-based and integrated systems gain higher utility, but cannot ensure this in the reputation and defenseless systems. We also find that the rate drops much faster in the integrated system than in the price-based system. The faster drop rate of integrated system is because the low-reputed nodes in the integrate system have higher incentives to be cooperative than in the price-based system because of the differentiated reputation-based prices. As the defenseless system has no mechanism to encourage cooperative behaviors or punish selfish behaviors, all nodes in the system are uncooperative. In the reputation system, since maintaining the reputation value only above the reputation threshold can maximize a node's utility, the packet drop rate increases and then stays at around 0.8 because the reputation threshold was set to 0.2. These results are in line with the density result in Figure 6.1, Figure 6.2, Figure 6.3 and Figure 6.4 and verify that the integrated system provides the strongest cooperation incentives.

To show the effectiveness of the integrated system in selfish node detection, we let a packet receiver drops the packet if its account value is greater than zero or its reputation value is above the threshold, and its reputation is then decreased by 0.1. Otherwise, the receiver forwards the packet and its reputation is increased by 0.1, and We randomly choose a node to function as a selfish node, count the number of interactions between the selfish node and other nodes during the simulation time, and measure the account value of the node corresponding to different numbers of interactions. When the selfish node's reputation falls below the threshold or its account value falls below zero, it is put onto a blacklist. All other nodes refuse to interact with the node in the blacklist. We consider two kinds of selfish nodes: wealthy and silly selfish nodes, and wealthy and clever selfish nodes.

Figure 6.13 shows the account value of the selfish node in the price system and integrated system. We initially assign $10,000$ and 1000 credits to the selfish node to determine the systems' effectiveness in detecting the selfish node when it is wealthy and not wealthy. In the figure, "Integrated-1000" represents the scenario of the integrated system and 1000 initial credits. The same notation applies to other credits. When the initial credits are 1000, the selfish node's account value becomes 0 after 20 interactions in the price system and after eight interactions in the integrated system respectively; thus, the integrated system takes much less time to detect the selfish node. The integrated system reacts faster because the forwarding price in the integrated system is determined by the source node's reputation instead of staying constant as in the price system. As the reputation of the selfish node decreases, it must pay more for packet forwarding service. Therefore, the selfish node will run out of credits faster in the integrated system than in the price system.

The figure also shows that when the initial credits are $10,000$, i.e., when the selfish node is wealthy, its account value decreases very slightly in the first 20 interactions. According to this decrease rate, it will take a significantly longer time for the price system to detect the selfish and wealthy node based on its account value. The integrated system detects the

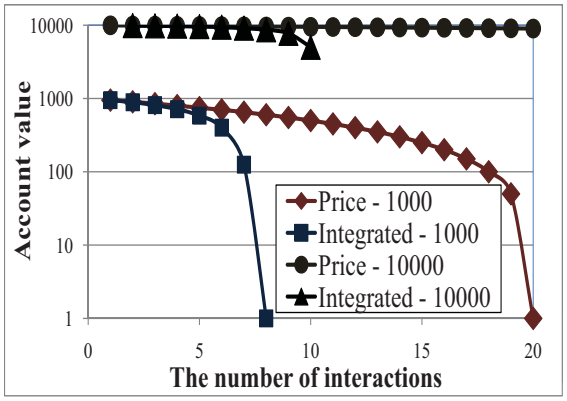


Figure 6.13: Detection of silly selfish nodes.

selfish node after only nine interactions since the reputation value of the selfish node falls below the reputation threshold even though its account value is still high.

A clever, selfish and wealthy node tries to maintain its reputation just above the reputation threshold to avoid being detected. Figure 6.14 shows the account value of such a node with $10,000$ credits in the integrated system, the reputation system, and the price system. Its account value in the reputation system is maintained at $10,000$ since it does not need to pay a price for packet forwarding. Because it can maintain its reputation value at the reputation threshold, the selfish node cannot be detected in the reputation system. The account value of the node drops slowly in the price system, and much more sharply in the integrated system. For the price system, since the selfish node has a large amount of initial credits, it takes a long time (i.e., more than 200 interactions) to be detected via account starvation. In contrast, the integrated system can detect the selfish node within only 40 interactions according to account starvation. The integrated system is more efficient because when the selfish node's reputation is at the threshold 1/5, it must pay a price five times higher than in the price system for each forwarding service. Therefore, its credits are consumed more quickly even though it is initially quite wealthy. The experimental results verify that the integrated system is more effective in detecting selfish nodes even when they are wealthy and clever.

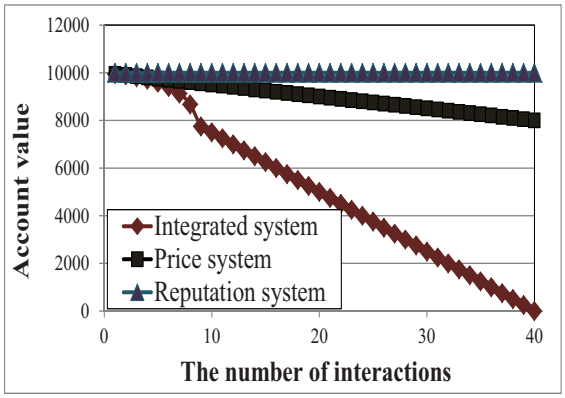


Figure 6.14: Detection of clever selfish nodes.

We further investigate the impact of the reputation threshold, the number of interactions, and the forwarding price on the account value of the selfish node with 1000 initial credits. In Figure 6.15, "Integrated-50" represents the integrated system with a packet forwarding price equal to 50 credits, which is applicable to other notations. The figure shows that at a certain reputation threshold and the same forwarding price, the account value decreases faster in the integrated system than in the price system. This decrease is due to the adaptive forwarding price based on reputation in the integrated system and constant forwarding price in the price system. The result confirms that the integrated system can detect selfish nodes more quickly. Comparing the results of "Integrated-50" with "Integrated-100", we observe that "Integrated-100" decreases much more rapidly because a higher forwarding price leads to a more rapid account value decrease.

The figure also shows that as the reputation threshold decreases, the account value drops more rapidly for both "Integrated-50" and "Integrated-100". Since a clever selfish node has a high incentive to maintain its reputation value around T_R, a low T_R will lead to a low stable reputation value for the selfish node. The selfish node then consumes its account value more quickly due to the reputation-adaptive forwarding price. In addition, we observe that the reputation threshold does not affect the account value in the price system because the system does not consider reputation. We also observe that if a low-reputed node

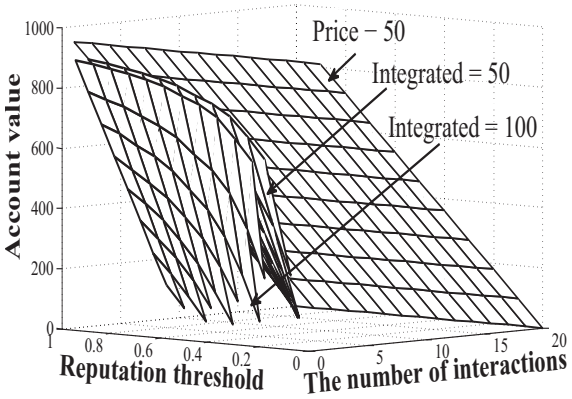


Figure 6.15: Account value vs. reputation threshold and number of interactions.

forwards a packet, its packet forwarding price decreases much faster than a high-reputed node. Therefore, in the integrated system, low-reputed nodes are highly encouraged to be cooperative.

Based on these results, we can conclude that compared to the reputation system and price system, the integrated system can more effectively defect selfish nodes.

6.2 Evaluation of the ARM System

We conducted simulations with NS-2 [4] to demonstrate the performance of ARM. We used the Distributed Coordination Function (DCF) of IEEE 802.11 as the MAC layer protocol. We chose the two-ray propagation model as the physical layer model, and the constant bit rate as the traffic mode. We describe our default settings below unless otherwise specified. The simulated network has 60 wireless nodes randomly deployed in a field of 1200$\times$ 1200 square meters. We randomly selected 10 nodes as managers. The radio transmission ranges of low-power and high-power interfaces were set to 250m and 1000m, respectively. The raw physical link bandwidth was set to 2Mbits/s. The heights of antennas for data transmitting and receiving were set to 1.5 meters. We used the random way-point mobility model [2] to generate node movement. The nodes are i.i.d. deployed in the field. They move

at a speed chosen from [1,10]m/s, wait for a pause time randomly chosen from [0,10]s, and then move to another random position. We randomly chose 10 pairs of source and destination nodes every 40s. The range of the reputations was set to [0,1], and the reputation threshold $\mathcal{T} = 0.4$. The deviation thresholds are set as δ_l=δ_g=δ_a=0.2. Each simulation lasted 5000s. We conduct 10 simulations and reported the average.

We set $\alpha = 0.7$ in Formula (4.8), $\varphi = 0.5$ in Formula (4.9), $p_r = 2$ in Formula (4.10), and $\gamma = 1$ in Formula (4.11). The time period T for periodically reporting information for mobile nodes and managers was set to 10s and 50s, respectively. Each node initially was assigned 5000 credits and a reputation value of 1. We compared the performance of the DSR [29] routing algorithm in the following systems: i) a defenseless MANET with neither reputation system nor price system (*Defenseless*), ii) a MANET with ARM, iii) a MANET with a representative reputation system (*Reputation*) [67], and iv) a MANET with a representative price system [37] (*Price*). To make the results comparable, rather than using the absolute number of forwarded packets, we use $R_l = \frac{D^r}{D^f}$ to evaluate a node's reputation value in *Reputation*. Selfish nodes maintain their reputation just above $\mathcal{T}$. In routing, a node chooses a node not on its blacklist for data forwarding. By default, every node just has one reputation manager.

6.2.1 Comparison of Performance of Different Systems

This experiment measures the system throughput with a certain fraction of uncooperative and reputed nodes, in which these selfish nodes keep their reputation just above the reputation threshold.

Figure 6.16 plots the average system throughput of different systems versus the percent of selfish nodes. The figure shows that ARM generates a higher throughput than *Reputation*, which produces a higher throughput than *Defenseless*. In *Defenseless*, a selfish node drops all of its received packets. *Reputation* can force selfish nodes to be cooperative to

a certain extent. However, a selfish node can still keep R_g just above $\mathcal{T}$ by dropping received packets with probability $\mathcal{T}$. In ARM, selfish nodes finally do not have enough credits to pay for their transmission services and are put onto blacklists. Therefore, ARM produces higher throughput than *Reputation*. Also, the figure shows the throughput of the system decreases as the number of selfish nodes grows. Since *Defenseless* and *Reputation* cannot detect all selfish nodes, their throughput decreases as the fraction of selfish nodes grows. It is intriguing to see that ARM also exhibits performance degradation though it can detect most selfish nodes. Such degradation is due to the reason that selfish nodes may be chosen as forwarding nodes before their credits are consumed. Also, avoiding selfish nodes in routing leads to longer path lengths, which suffers from higher transmission interference.

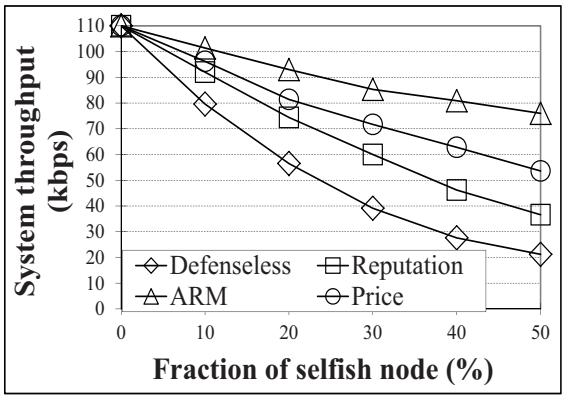


Figure 6.16: Average system throughput

To verify the effectiveness of punishing selfish nodes by refusing their transmission requests, we tested the throughput of packets generated by selfish nodes over a time interval. We setup 10 selfish nodes and used them as source nodes. Figure 6.17 plots the throughput of the selfish nodes. In *Defenseless*, selfish nodes maintain a constant throughput of 15kbps. In *Reputation*, the throughput decreases as time elapses and then remains constant at 6kbps because the selfish nodes keep R_g just above $\mathcal{T}$, thus their transmission requests are accepted by other nodes. The throughput in ARM declines sharply over time and finally reaches 0. That is, with the aid of account management, ARM can effectively detect and punish selfish

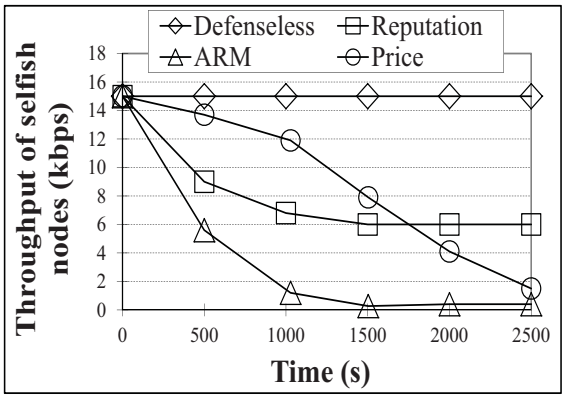


Figure 6.17: System overhead

nodes, excluding them from the network.

To evaluate the efficiency of the systems, we tested the overhead measured in kbps for all overhead messages in the systems. In addition to the "hello" messages, the overhead messages in ARM also include those for topology construction and maintenance, and reputation querying; in *Reputation*, the overhead also includes the messages for reputation exchange; in *Price*, the overhead also includes the messages for credit payments. Figure 6.18 illustrates the overhead in each system versus network size. The figure demonstrates that ARM yields much less overhead than *Price*, which produces less overhead than *Reputation*. In ARM, since nodes only communicate with managers, the overhead is proportional to the network size. Though ARM must construct and maintain the DHT infrastructure in node mobility, its total overhead is still lower than the other systems. In *Reputation*, the reputation information is exchanged among local nodes periodically, resulting in much higher overhead. In *Price*, credit circulation in the network generates transmission overhead. The results confirm that ARM consumes less resource than reputation and price systems.

6.2.2 Evaluation of the DHT Infrastructure in ARM

We measured the average, maximum, and minimum connectivity degree per manager when the managers move at the speed of 1m/s, 10m/s, and 20m/s. In addition to the default

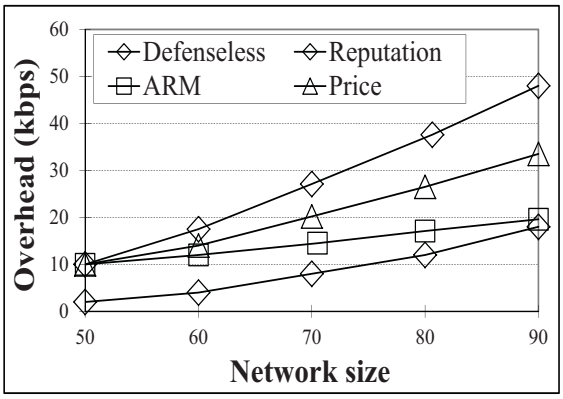


Figure 6.18: Performance comparison between different systems

experiment scenario with 10 managers, we also measured the performance with additional 10 and 20 managers, respectively.

Figure 6.19 shows that the smallest connectivity degree of a manager is about $\frac{N}{2}$. The figure also shows that more managers incur higher connectivity degree because a manager has more neighbors in a DHT with more nodes. We find that the node mobility does not affect the connectivity degree per manager, thus the proposed DHT maintenance mechanism can establish new links immediately upon link breakups.

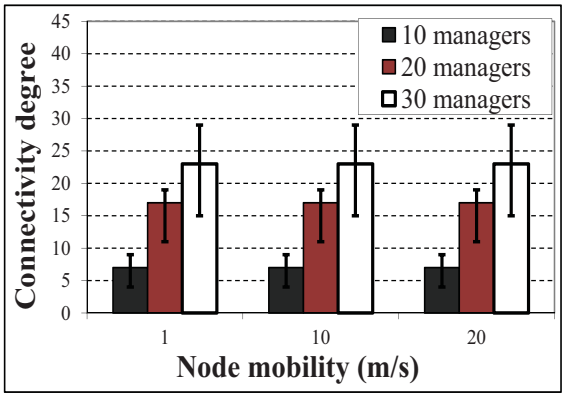


Figure 6.19: Connectivity degree per manager

Figure 6.20 presents the average connection duration of managers versus node mobility. We also include the theoretical results based on Proposition 4.2.2 in the case of "10 managers". The figure demonstrates that when the mobility is 0.5 m/s, the DHT in-

frastructure is much more stable than other situations. As node mobility increases, the average connection duration drops sharply. We also find that because the high power interface permits a manager to contact a manager within a long range, the connection duration is nearly identical. Thus, the number of managers does not greatly affect the stability of DHT infrastructure. The simulation results closely match the theoretical results as shown in Proposition 4.2.2. The small gap between the simulation and theoretical results is because the theoretical analysis does not consider the node pause time during movement.

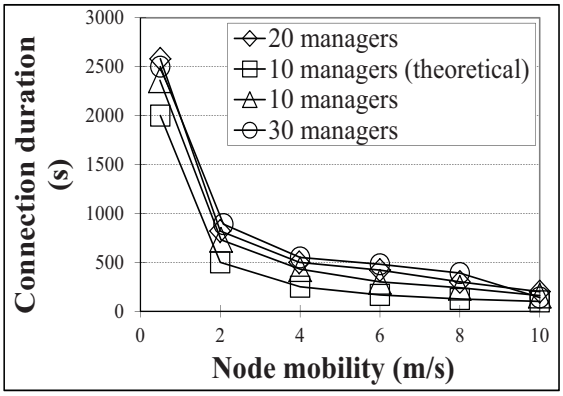


Figure 6.20: Average connection duration

Figure 6.21 shows the maintenance overhead of the DHT infrastructure versus node mobility. The overhead is represented by the number of messages exchanged for DHT maintenance and grows with the increase of node mobility. Higher mobility leads to higher probability of link breakups, incurring higher maintenance overhead. The overhead also grows as the number of managers increases, because more managers generate more messages for DHT maintenance. Therefore, fewer nodes with low mobility should be chosen as managers to reduce DHT maintenance overhead.

Figure 6.22 shows the number of nodes that have been reassigned IDs in DHT maintenance over time, specifically indicating that most often, two nodes need ID reassignment for DHT maintenance. Although the physical link between nodes n_i and n_j is broken, they still share the same manager neighbors. Therefore, by re-ordering the IDs of n_i and n_j

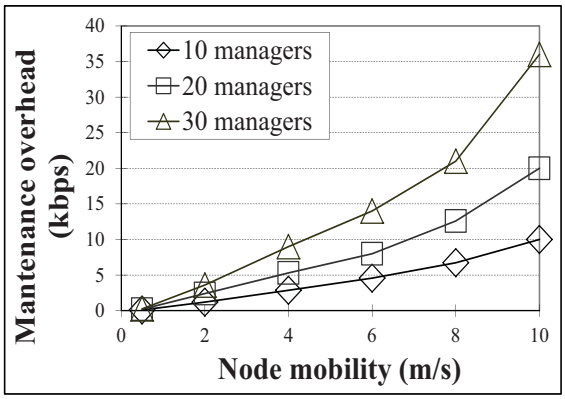


Figure 6.21: Topology maintenance overhead

and the shared neighbors, the DHT structure with numerically continuous IDs can often be recovered.

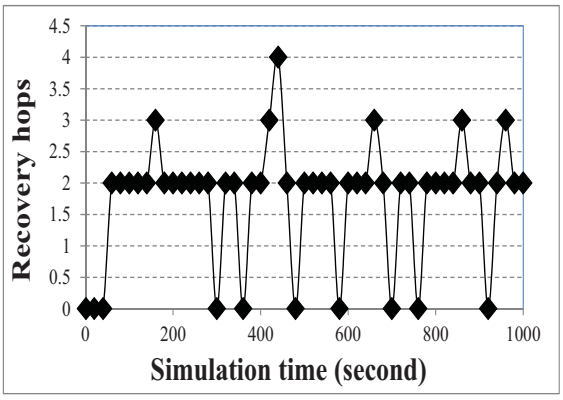
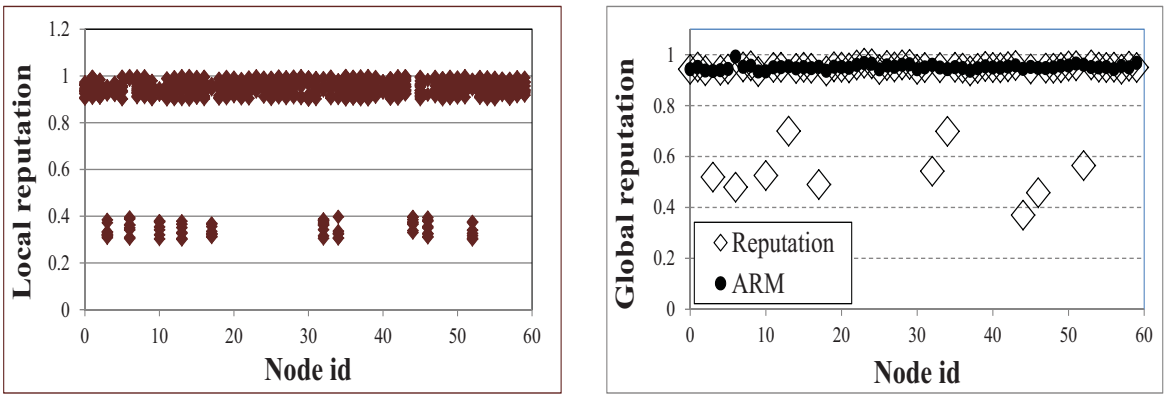


Figure 6.22: Reassigned IDs in DHT maintenance

6.2.3 Evaluation of Misreport Resilience

We also tested whether ARM can accurately calculate node reputation with misreports due to an adverse environment with interfering background noise. We increased the background noise in 10 randomly chosen regions. All nodes in the system are cooperative. Figure 6.31a shows node local reputations in the defenseless system. The low reputations are caused by misreports due to background noise. It is very interesting to find that the nodes with low reputation are clustered. The reason is that in the adverse environment, physically



(a) Defenseless system (b) Defensive system

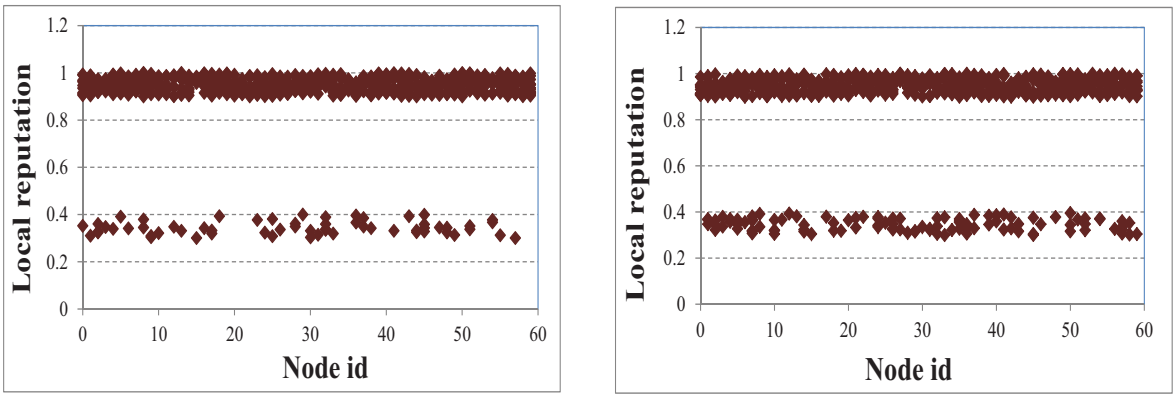
Figure 6.23: Node reputation in adverse environment

close nodes experience the interference noise at the same time. Figure 6.31b shows ARM can accurately reflect nodes' reputation in the adverse environment while *Reputation* cannot accurately reflect some nodes' reputations. ARM is able to collect all reports in the system by relying on its DHT infrastructure, identify the cooperative nodes in the adverse environment by analyzing the clustering features of the nodes with low reputations, and then filter their reports.

6.2.4 Evaluation of False Accusation Resilience

In this experiment, all nodes are cooperative. We select some nodes that will deliberately evaluate their neighbors with low reputations randomly chosen between $[0.3, 0.4]$.

In the defenseless system, every node rates its neighbors based on their forwarding behavior. Figures 6.24(a) and (b) show the plot of all evaluated local reputations of each node in a defenseless system with five and ten false-reporting nodes, respectively. Because of the false-reports, some of the cooperative nodes are rated with low reputations. Comparing Figures 6.24(a) and (b), we see that as the number of the false-reporting nodes increases, the number of low reputations each node received increases.



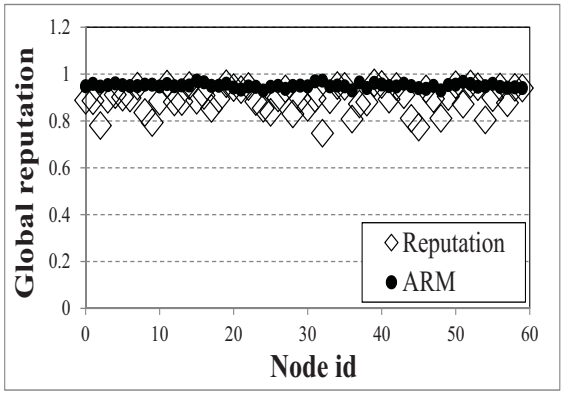
(a) Five false-reporting nodes

(b) Ten false-reporting nodes

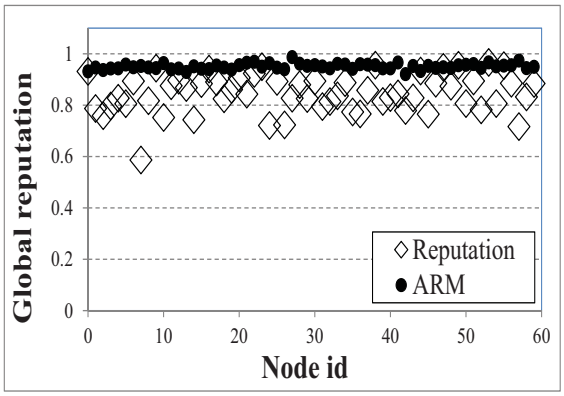
Figure 6.24: Local reputations in a defenseless system.

To make the global values of a given node in different nodes identical, we used broadcasting to ensure that each node receives the local reputations from others. Figure 6.25 shows the global reputation of each node in *Reputation* and ARM. *Reputation* exhibits a large variance in reputations and cannot accurately reflect the reputations of these nodes. The reason is that in *Reputation*, each node considers the false reports when calculating the global reputations. In the figure, all reputations in ARM are close to 1, which means ARM can more accurately reflect node reputations. Some reputations are not 1 because some cooperative nodes may drop packets due to transmission interference.

Comparing Figure 6.25(a) with Figure 6.25(b), we find that more false-reporting nodes in the system generate greater variance in node reputations in *Reputation*, while they exert no significant influence on node reputations in ARM. More false reports incur a greater inaccuracy of node reputations in the final global reputation calculation in *Reputation*. In contrast, by taking advantage of the DHT infrastructure, ARM can efficiently gather all local reputations of each node in the system and filter the false reports.



(a) Five false-reporting nodes



(b) Ten false-reporting nodes

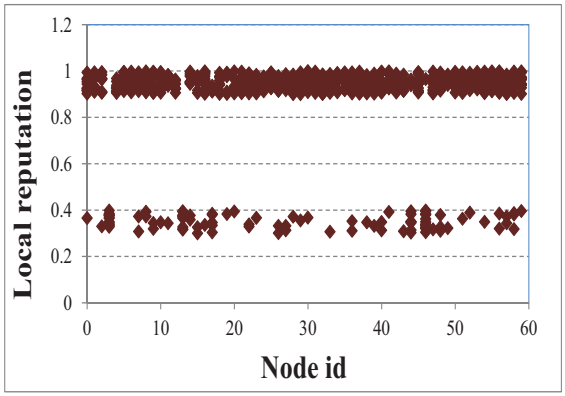
Figure 6.25: Node reputations in a defensive system.

6.2.5 Evaluation of Collusion Resilience

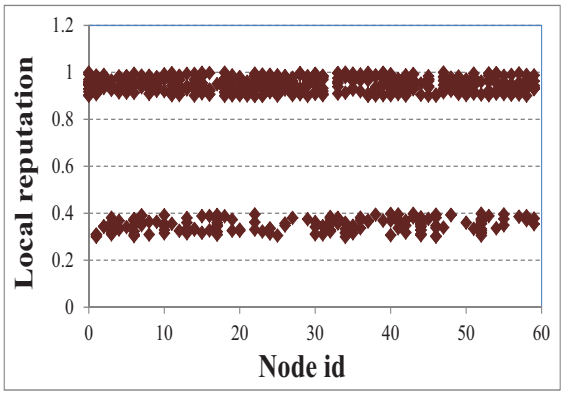
According to the movement of the colluders, collusion is classified as both non-group collusion and group collusion. In the former, the colluders move individually, and they report a high reputation for each other when meeting together. In the latter, all colluders in a group move together as a group, and always rate high reputations for each other. We conducted experiments for both non-group collusion and group collusion. We consider collusion in which colluders drop received packets with probability 0.3, and falsely report low reputation randomly chosen between [0.3,0.4] for their neighboring cooperative nodes, and higher reputation randomly chosen in [0.9,1] for other colluders.

Figures 6.26(a) and (b) show node local reputations in a defenseless system with 5 and 10 colluders, respectively. The figures show that a certain portion of nodes receive low R_ls that are from the false reports of the colluders on benign nodes and correct reports from benign nodes on colluders. Comparing the two figures, we find that the number of low R_ls is proportional to the number of colluders in the system.

Figures 6.27 (a) and (b) show the global reputation of each node in *Reputation* and ARM in non-group collusion. *Reputation* exhibits a larger variance than ARM in the repu-



(a) Defenseless system (5 colluders)

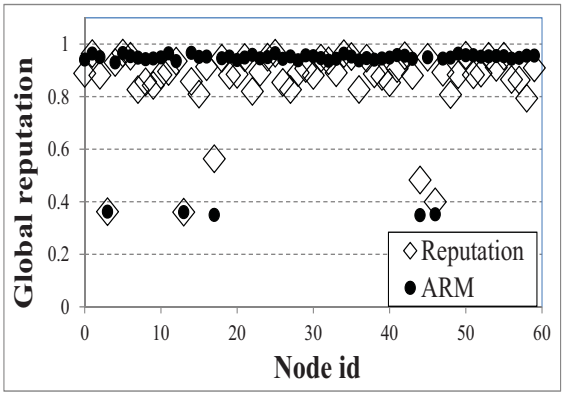


(b) Defenseless system (10 colluders)

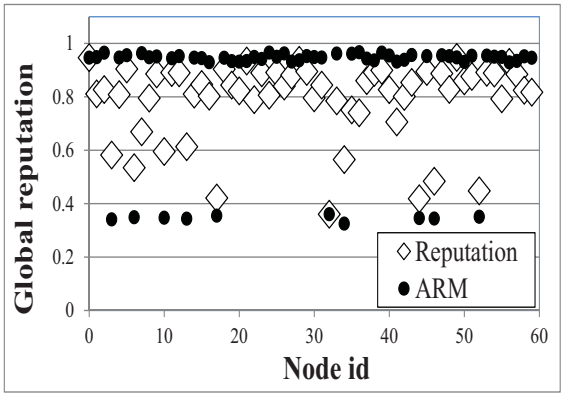
Figure 6.26: Reputations in a defenseless system with non-group collusion.

tations of cooperative nodes, especially when the number of colluders increases. The reason is that *Reputation* includes the false reports for the cooperative nodes in calculating global reputation. More colluders lead to more false reports. By collecting all reports in the system through the DHT infrastructure, ARM can easily identify and filter the reports from colluders that are largely different from others, since the majority of the nodes in the system are benign. Also, though both systems can identify the colluders, *Reputation* cannot accurately reflect the R_gs of colluders since some colluders have high R_gs. The reason is that the colluders report high R_ls for each other when meeting each other. *Reputation* considers these false reports while ARM filters them when calculating R_g.

Figure 6.28 shows the local reputations for group collision in a defenseless system. Compared to Figure 6.26, Figure 6.28 has fewer low node R_ls for two reasons. First, in the group node collusion, the colluders can always report high reputations for each other to increase their own reputations. Second, more colluders generate more low R_ls for cooperative nodes. Figures 6.29(a) and (b) show the global reputation with group collision in *Reputation* and ARM. When the number of colluders is five, even though they always collude with each other, *Reputation* and ARM can identify the colluders since the majority of the neighbors



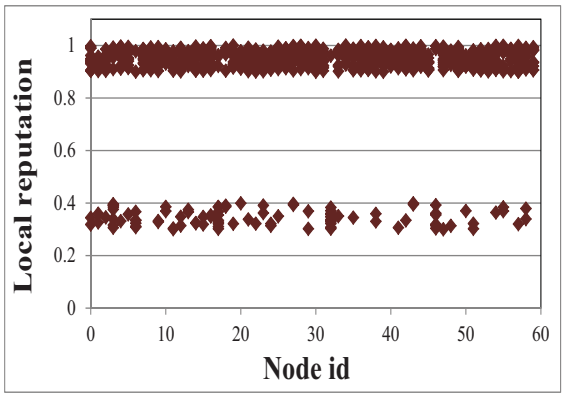
(a) Defensive system (5 colluders)



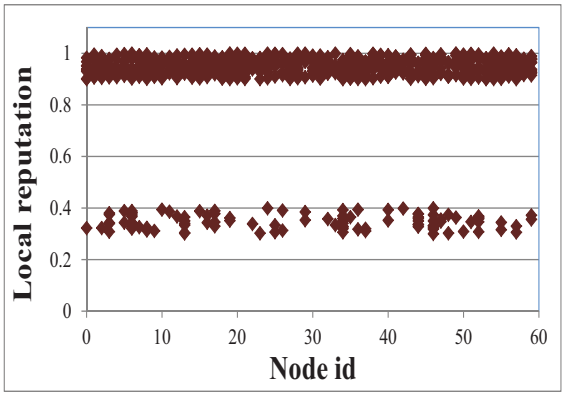
(b) Defensive system (10 colluders)

Figure 6.27: Reputations in a defensive system with non-group collusion.

of a colluder are benign. By filtering the false reports, ARM generates more accurate R_gs than *Reputation* for both cooperative nodes and colluders. However, when the number of the colluders increases to ten, it is very difficult for *Reputation* to detect colluders. Also, ARM cannot detect some colluders directly based on reputation, since the majority of the neighbors of a colluder are colluders and the false reports from the colluders overwhelm the reports from benign nodes. The account management in ARM can help detect the colluders as shown in Figure 6.30.

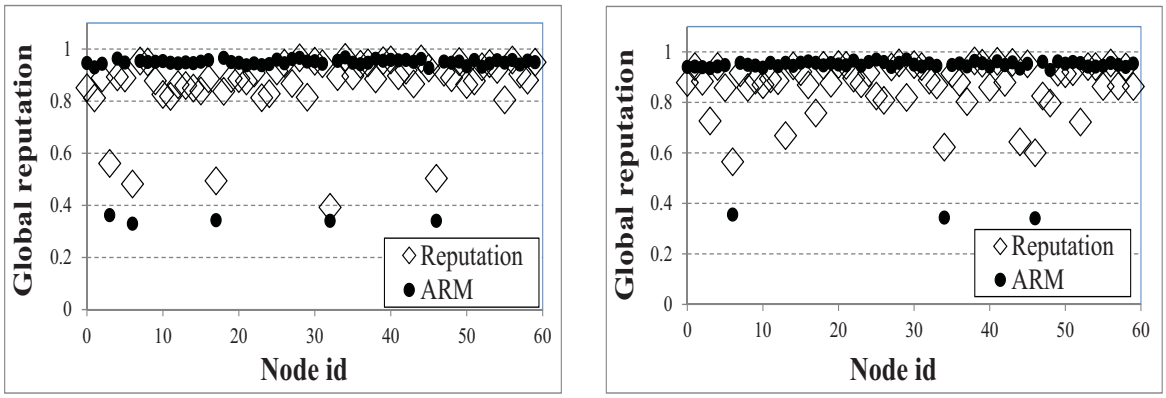


(a) Defenseless system (5 colluders)



(b) Defenseless system (10 colluders)

Figure 6.28: Reputations in a defenseless system with group collusion.



(a) Defensive system (5 colluders) (b) Defensive system (10 colluders)

Figure 6.29: Reputations in a defensive system with group collusion.

Figure 6.30 shows the account value of each colluder versus the number of generated packets in ARM. We can observe that the colluders' account credits decrease linearly as they generate more packets. Although the colluders can maintain a high R_g by rating each other high and receiving fraudulent benefits of a low service price, they will ultimately consume their credits as they generate more packets, and finally are detected as uncooperative nodes by deficit accounts.

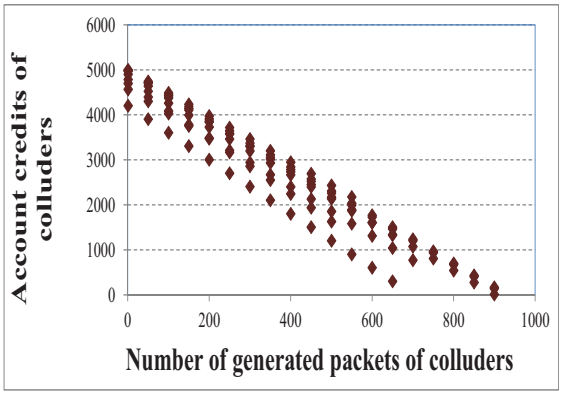
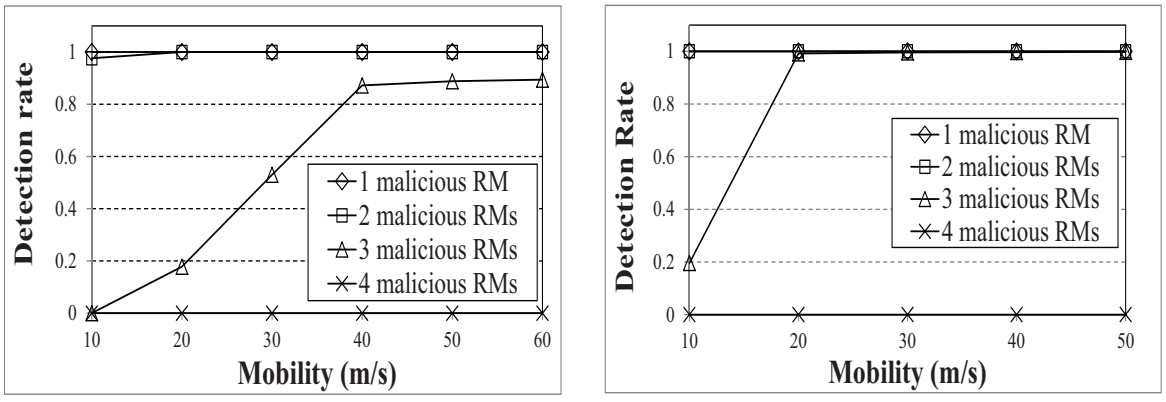


Figure 6.30: Credits of colluders.

6.2.6 Evaluation of Reputation Manager Auditing

In this section, we test the performance of the reputation manager auditing mechanism. We set a different number of malicious reputation managers in the system as either local reputation or global reputation collection. We then ran twenty simulations and report the average.

Figure 6.31 shows the malicious reputation manager detection rate in a local reputation calculation with varying number of malicious reputation managers in the system. We see that the reputation manager auditing algorithm can effectively detect the malicious reputation managers when the number of malicious reputation managers is small. In this situation, the probability that a node meets a malicious reputation manager during a certain time period is low. However, as the number of malicious nodes in the system increases, the probability that a node meets a malicious manager increases. More malicious managers report false reputations for a node, leading to a decreased detection probability. When the number of malicious reputation managers in the system is very large, however, a node has a high probability of meeting many malicious reputation managers with a detection rate that is very low.



(a) Update period = 10s (b) Update period = 20s

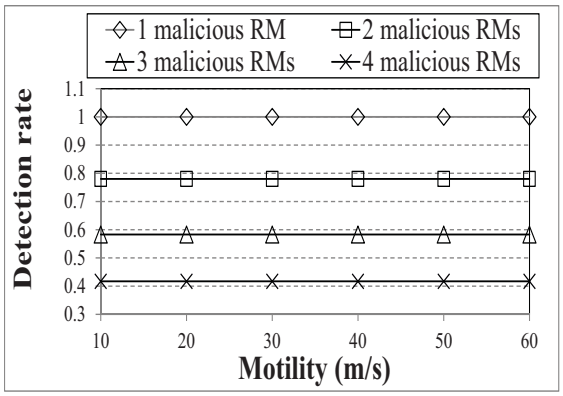
Figure 6.31: Malicious reputation manager detection in local reputation collection.

We can also see from the figure that as the average mobility of the nodes in the system increases, the probability of detection increases. Higher mobility enables a node to meet more reputation managers in a reputation update period, which helps detect malicious reputation managers based on reputation reports from more total reputation managers. Comparing Figure 6.31 (a) and Figure 6.31 (b), we find that as the reputation update period increases from 10s to 20s, the detection rate increases. A longer reputation update period enables a node to meet more reputation managers during the period, which helps detect malicious reputation managers based on reputation reports from more total reputation managers.

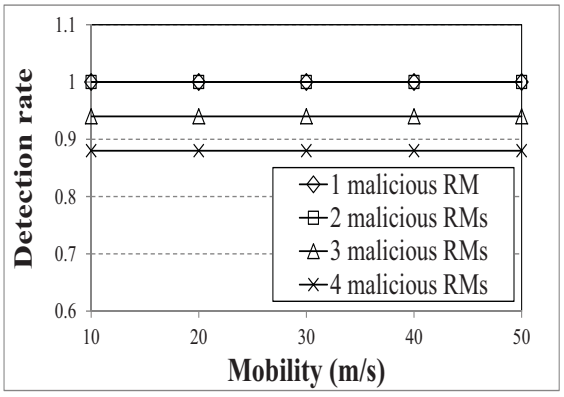
Figure 6.32 shows the malicious reputation manager detection rate in global reputation calculation versus node mobility with varying number of malicious nodes in the system. We see from the figure that, as the number of malicious reputation managers increases, the malicious reputation manager detection rate decreases. The increased number of the malicious reputation managers increases the probability that a node has a malicious owner manager. We also see that node mobility does not affect the detection rate because malicious reputation managers are detected by the deviation of reported information, which is not affected by node mobility. Comparing Figure 6.32 (a) and Figure 6.32 (b), we find that as the number of reputation managers of one node increases, the malicious reputation manager detection rate increases. Such increase is because given a constant number of malicious reputation managers in the system, as more reputation managers manage the reputation value of a node, the ratings from malicious reputation managers can be more easily identified.

6.3 Evaluation of the SEDUM Routing Protocol

This section demonstrates the distinguishing properties of SEDUM through simulation on "The ONE" simulator [53] in comparison with Epidemic routing [94] (denoted by Epidemic), and *Spray and wait* routing [90] (denoted by SW) based on the basic community



(a) Num. of RMs of a node c=2



(b) Num. of RMs of a node c=4

Figure 6.32: Malicious reputation manager detection in global reputation collection.

model and the Manhattan community model. "The ONE" simulator is specifically designed for evaluating DTN routing and application protocols written in Java. In Epidemic, when two nodes n_i and n_j meet each other, n_i copies to n_j its messages that n_j has never received before. In SW, a source node replicates a certain number of replicas to its neighbor nodes in the system. The replica nodes buffer the replicas until meeting the destination node. We also compared the frequency utility based SEDUM and duration utility based SEDUM. The experimental results confirm that the duration utility produces higher throughput and lower delay than the frequency utility.

We used two node movement models (Figure 5.2): the basic community model and the Manhattan community model. In the experiments, 150 nodes are i.i.d. in a $2000m \times 2000m$ area. We assigned 15 interest points including 7 home communities and 8 gathering places, and randomly chose 10 nodes to share an interest point to show the colocation node movement pattern. The home communities and gathering places are randomly distributed in the area. Unless otherwise specified, the moving speeds of mobile nodes were randomly chosen from (0-20]m/s. Each node has a 40m transmission range and a 40Mb buffer size. At every second, two nodes are randomly chosen to generate a new message with a size of 1Mb for a randomly selected destination node with a transmission rate of 2Mb/s for 2000

seconds. We assigned the same priority of tolerable delay to all messages. Initially, each node randomly chooses three points as its interested points and is assigned a probability to visit each of these three points. The probability reflects the likelihood that a node will move to the point. To move from one interest point to another, a node chooses the shortest path based on the Dijkstra shortest path algorithm. The Basic community model imposes no node movement restrictions. The Manhattan community model confines the routing paths of the mobile nodes to certain paths that reflect their real moving pattern in addition to the colocation pattern. The Manhattan model consists of grids in a matrix, in which all nodes can only move on the sides of a grid.

We set a 5-hop Time to Live (TTL) for messages in the three protocols. The number of replicas of a message in SEDUM and SW was set to eight. When a node is in a home community, it will go to a gathering place with a probability of 0.8, and will go to other randomly chosen places with a probability of 0.2. When a node is in a gathering place, it will then go home with a probability of 0.5, and go to other places with a probability of 0.5. After reaching a point of interest, the node will stay there for a time period randomly selected between [10-15]s. When a node is at other places, it will go back to its home community directly.

All experimental results were averaged over ten runs. A warm up period of 500s is used at the beginning of the simulations to initialize the utility of SEDUM. The simulation time is 4000s. We use SEDUM-20 and SEDUM-40 to represent nodes in SEDUM with transmission range of 20m and 40m, respectively. It is similar to SW-20, SW-40, Epidemic-20, and Epidemic-40. We are mainly interested in four metrics in the simulation:

(1) Message delivery delay: the average time period that it takes a message to be delivered to its destination;

(2) Message delivery capability: the total number of messages that are delivered to the destinations;

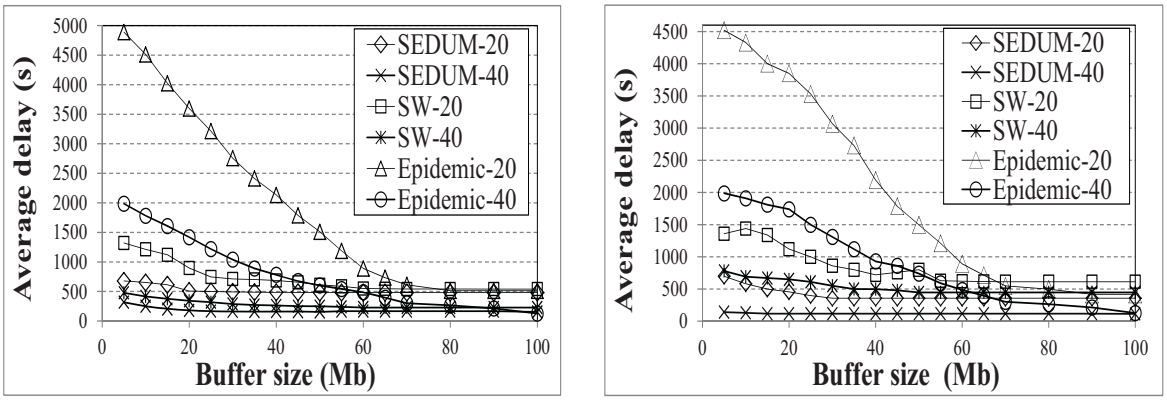
(3) Message delivery overhead: the total amount of traffic needed to deliver the messages, including control traffic (e.g. delivered message list and exchanged matrix table) and replica exchanging traffic;

(4) Success rate: the ratio of the number of successful delivered messages to the total number of initiated messages.

6.3.1 Evaluation of Message Delivery Delay

Figures 6.33 (a) and (b) show the average message delivery delay versus the node buffer size in the Basic community and Manhattan community models, respectively. From Figure 6.33, we observe that the delivery delay for all systems decreases greatly as the node transmission range increases from 20m to 40m because a larger transmission range enables a node to contact more neighbor nodes, which increases the probability of meeting the destination or nodes having a high utility for the destination. Both figures show that the average delivery delay decreases as the node buffer size increases. Epidemic shows a sharp drop, while SEDUM and SW show slight drops. A larger buffer size enables a node to buffer more messages in transmission, thus reducing the probability that a message is discarded due to buffer congestion. Because of the flooding, Epidemic produces many more message copies, which increases the frequency of congested buffers. Thus, many messages are dropped in small buffers, while large buffers enable nodes to store more messages in routing, thus greatly reducing the routing delay. When the buffer size of a node is large enough to store all of the messages in a system, a source node routes a message through the shortest path to the destination by flooding messages in the network. In this situation, the delay of Epidemic is the lower bound of the network's delay.

We can also see from both figures that SEDUM is delayed the least, although SEDUM replicates a message to several nodes just as SW does. SW uses direct routing with $O(\frac{N}{N_c})$ delay, in which messages are routed to their destinations merely by chance. SEDUM uses



(a) Basic community model (b) Manhattan community model

Figure 6.33: Delivery delay vs. buffer size.

probabilistic routing, in which the messages are routed to nodes with a higher probability of meeting their destinations. In addition, the buffer management mechanism in SEDUM further reduces delivery delay. SEDUM gives higher priority to both longer-lifetime messages for transmission in routing and to higher-utility messages to retain in a buffer when it is congested. Further, the multi-copy based probabilistic routing can increase the probability that a replica is forwarded to its destination through a relatively shorter path. Thus, SEDUM has lower message transmission delays than SW. As Epidemic relies on flooding, it suffers from severe buffer congestion because of limited buffer sizes and a tremendous number of messages. When the buffer size is small, buffer congestions cause many message drops, which cause the high delay in Epidemic. We also see that the delay of SEDUM is nearly identical to Epidemic with a large buffer size, which means that SEDUM can reach the lower bound of the delay performance of the network with a large buffer size.

Comparing Figures 6.33 (a) and (b), we find that the transmission delay of SEDUM in the Manhattan community model is lower than that in the Basic community model when the transmission range equals 40m. The nodes only move along the edges of the grids, thus increasing the probability that two nodes sharing similar movement patterns will meet. By

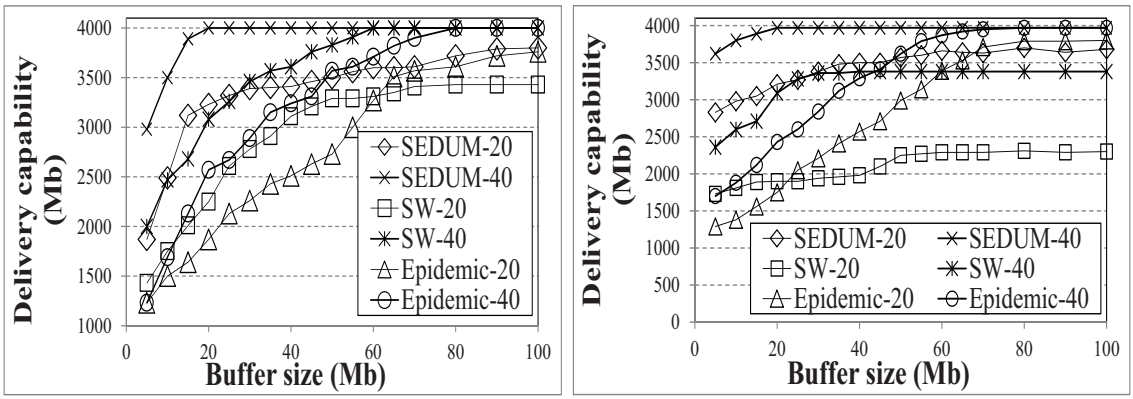
capturing the colocation and familiar stranger attributes of the node movement pattern, SEDUM's duration utility can accurately reflect the communication capacity of two nodes. Therefore, the messages are forwarded to the destination nodes through a number of relay nodes that share increasingly similar movement patterns with the destination.

In contrast, SW in the Manhattan community model incurs a higher delivery delay than in the Basic community model when the transmission range equals 40m. In SW, a source node replicates a message to a number of neighbor nodes. A message is delivered to the destination only when one replica node meets the destination. However, since the nodes move along certain movement paths in the Manhattan community model, it is very likely that no replica node can meet the destination node if the replica nodes belong to different communities. Therefore, the transmission delay of SW in the Manhattan community model increases. We notice that with a 20m transmission range, SEDUM generates similar delays in both models, as does SW. Because the transmission range is already small, different node movement models do not significantly change the number of nodes a node can contact. For Epidemic, the messages are flooded in the system with TTL, which is not affected by node movement patterns. Thus, its delay in both models remains approximately the same.

6.3.2 Evaluation of Message Delivery Capability

Figures 6.34 (a) and (b) show the total delivery throughput versus the buffer size for the Basic community and Manhattan community models, respectively. From both figures, we see that a larger transmission range increases the throughput of SEDUM, SW, and Epidemic. A shorter node transmission range reduces the probability of node contacts, thus reducing the number of successfully delivered messages. Figures 6.34 (a) and (b) also show that SEDUM produces a higher throughput than SW, followed by Epidemic. SEDUM forwards messages to nodes in longer contact with the destination, thus enabling greater message delivery capacity. Also, SEDUM has a buffer management protocol that gives higher-utility messages a higher

priority to remain in a congested buffer and to give longer-lifetime messages a higher priority to be sent out from a buffer. Without the utility and buffer management strategies, SW only relies on direct routing with a TTL in the forwarding phase. Each node holds message replicas until it either meets the destination node or the number of the message's forwarding hops exceeds the TTL. Messages must stay longer in the buffer to meet their destinations as the node transmission range decreases. Therefore, more messages are dropped as their TTL expires. As a result, SW suffers more than other routing protocols from a decrease of transmission range. Since Epidemic severely suffers from buffer congestion, especially under a high load due to flooding, the throughput follows SEDUM>SW>Epidemic.



(a) Basic community model (b) Manhattan community model

Figure 6.34: Delivery throughput vs. buffer size.

We also see from the figures that as the buffer size increases, so does the number of messages successfully delivered to their destinations. A larger buffer size means more messages can be buffered, and the probability that a message is thrown away from a buffer decreases. Therefore, the number of messages received by the destination nodes is increased. The figures also demonstrate that when the buffer is large enough for most messages, the throughput of SEDUM is comparable to Epidemic in a low-load network, indicating the high throughput performance for SEDUM.

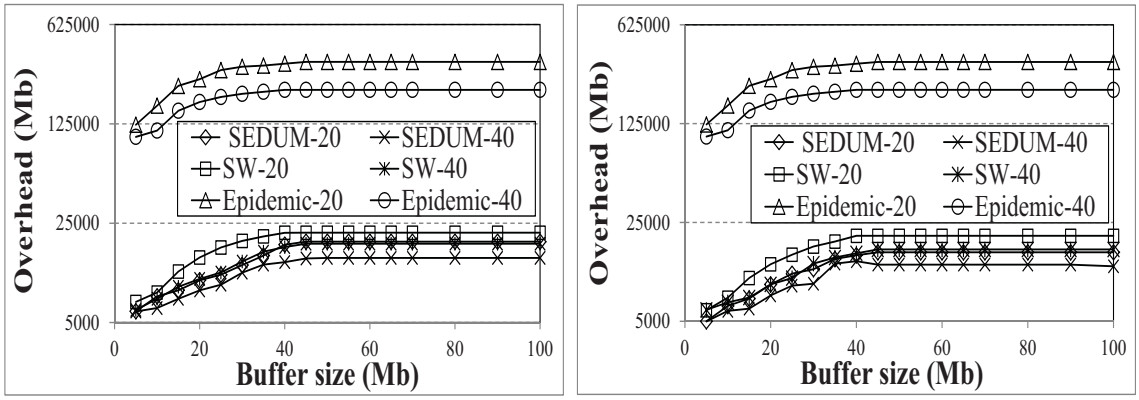
Comparing Figures 6.34 (a) and (b), we observe that SEDUM can deliver more mes-

sages in the Manhattan community model than in the Basic community model because nodes with similar movement patterns have a high probability of meeting each other in the Manhattan community model. Also, the duration utility that can capture the colocation and familiar stranger attributes of node movement patterns enables a message to travel along a path consisting of nodes sharing a similar pattern of movement with the destination. As this pattern expedites message delivery and avoids message congestion in node buffers, more messages can be successfully delivered to their destination nodes. In contrast, since the confined routing paths may reduce the meeting probability of a replica node with the destination node, the delivery rate of SW in the Manhattan community model decreases. In Epidemic, since a source always floods a message to the destination node, the number of received messages in the Basic community model and the Manhattan model remains nearly identical.

6.3.3 Evaluation of Message Delivery Overhead

Figures 6.35 (a) and (b) show the total overhead involved in message delivery in the systems versus the buffer size in Basic community and Manhattan community models, respectively. We see from the figures that as the node's transmission range increases from 20m to 40m, the system transmission overhead is reduced. Such reduction is because a large transmission range increases the chance of a node meeting the destination node in a short time, leading to a reduced number of replicas exchanged in the system.

Figure 6.35 also shows that SEDUM has smaller overhead than SW, followed by Epidemic. This is because the optimal tree replication and buffer management protocols enable SEDEM to deliver the messages to their destinations in a much a shorter time than SW and Epidemic, as shown in Figure 6.35, reducing the number of replicas exchanged in the system. We also see from the figures that as the buffer size increases, so does the overhead in transmission. When the buffer size is small, some of the replicas are dropped because of



(a) Basic community model (b) Manhattan community model

Figure 6.35: Delivery overhead vs. buffer size.

the congestion in the buffer. Therefore, the number of replicas exchanged in the system is reduced. However, such low overhead is achieved at the cost of the high message delivery delay and low message delivery capability, as shown in Figure 6.33 and Figure 6.34.

Comparing Figures 6.35 (a) and (b), we observe that SEDUM has slightly less overhead in the Manhattan community model than in the Basic community model. The less overhead is due to the reason that nodes with similar movement patterns have a high probability of meeting each other in the Manhattan community model. Therefore, it takes fewer replica exchanges to deliver a message, which leads to less overhead. In Epidemic, since a source always floods a message to the destination node, the amount of overhead in both models is almost identical. We also see from the figure that the overhead in SW for the Manhattan model increases slightly as no replica node can meet the destination node if the replica nodes belong to different communities. Therefore, the number of replica exchanges in SW for the Manhattan community model increases.

Figure 6.36 shows a further investigation into the message delivery overhead for SEDUM. We see from the figure that compared to the replica overhead, the amount of the control overhead in SEDUM is negligible. The reason is that although the nodes in the system must exchange their utility table and delivered message list constantly, the size of

the metadata in these tables and lists are in Kbytes, which is so small that the amount of control overhead imposes negligible effects on system performance compared to the data messages. We also see from the figure that both the control overhead and the replication overhead in the Manhattan community model is smaller than that of the Basic community model. As shown in Figure 6.33, the message delivery in the Manhattan model is faster than the Basic community model, a reduced number of replicas is buffered in each node, reducing the amount of traffic in exchanges of replicas, utility tables, and delivered message lists.

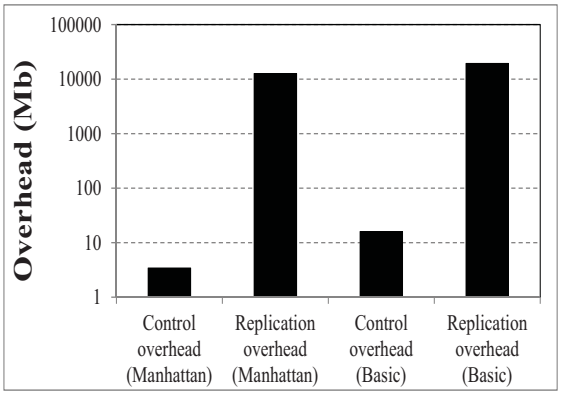


Figure 6.36: Overhead in SEDUM

6.3.4 Evaluation of the Effect of the Number of Replicas on Delay

In this experiment, we varied the number of replicas per message from 2 to 10 with an increase of 2 in each step, and measured the cumulative distribution function (CDF) of message delivery delay in the Manhattan community model, as shown in Figure 6.37. In the figure, "250+" means that the message delivery delay is larger than 250s. We can see from the figure that as the number of replicas per message increases, more messages can be delivered in a short time. However, the delay decrease rate is not proportional to the number of replicas per message. As we can see, the message delivery delay with 8 replicas per message is slightly less than that with 10 replicas per message. The result is in line with Theorem 5.3.1.

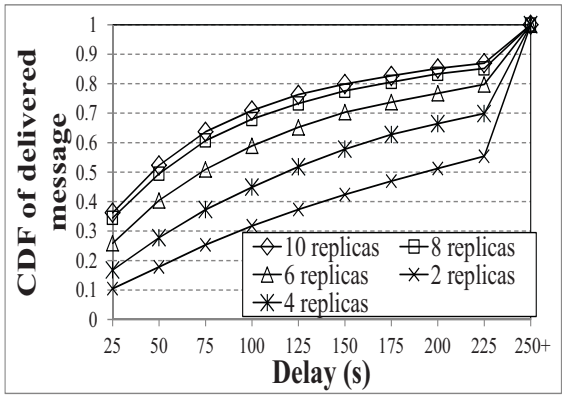


Figure 6.37: Delay vs. number replicas.

6.3.5 Comparison of Different Replication Methods

In this section, we compare the performance of SEDUM for the Manhattan community model using different replication methods: optimal tree replication, source tree replication, and binary tree replication. Table 6.1 shows the comparison results of the methods in terms of the average overhead, the average delay, and the average throughput. We see from the table that the source tree replication method leads to the largest amount of overhead, the longest average delay, and the smallest average throughput. This is because the source tree replication method is the slowest replica replication as only the source node is allowed for the message replication. Such requirement reduces the chance for a replica to meet the destination node. The longer the time for a message to be delivered, the longer the time the message must stay in the buffer, which may result in congestion and reduce the throughput. Meanwhile, the longer the time a message stays in the buffer, the higher the chance of the message being exchanged among nodes, which can result in a high delivery overhead. Since in binary tree replication, every node holding the replica is allowed to forward the replicas to 2 neighboring nodes, the performance of the binary tree replication is better than the source tree replication. Since the optimal tree replication method is the fastest replication method as explained in Section 5.2.3, the optimal tree replication has the highest performance.

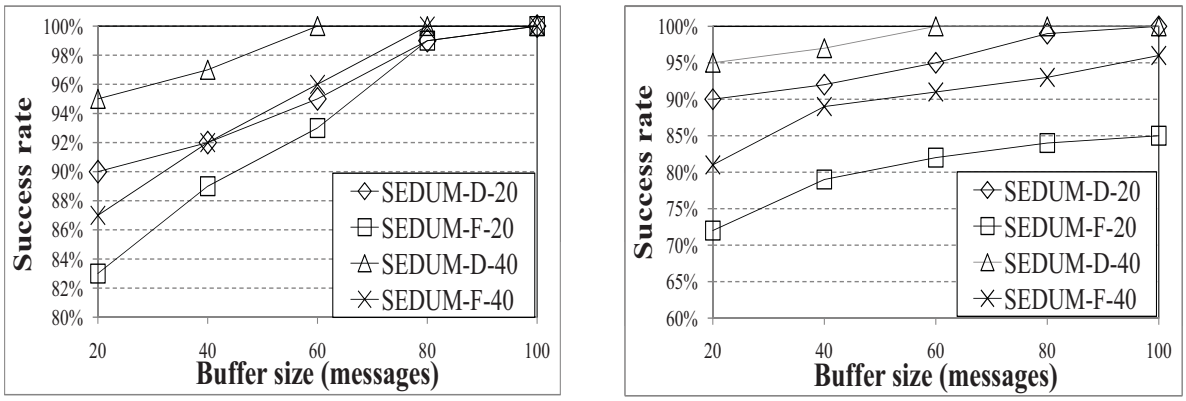
Table 6.1: Comparison of different replication methods

	Optimal tree	Source tree	Binary tree
Ave. overhead:	16112.4Mb	17514.4Mb	16943.8Mb
Ave. delay:	117s	213s	141s
Ave. throughput:	3970Mb	3714Mb	3873Mb

6.3.6 Comparison of Frequency Utility and Duration Utility

In this section, we compare the frequency utility based routing and the duration utility based routing to verify our analytical results in Section 5.1. We use the Manhattan community model to simulate the movement of nodes since this model is more realistic. Figures 6.38 (a) and (b) show the success rate of SEDUM using the duration utility and the frequency utility with 0.4Mb and 4Mb message sizes, respectively. In the figures, SEDUM-D-20 denotes SEDUM using the duration utility and a 20m node transmission range. SEDUM-F-20 denotes SEDUM using the frequency utility and a 20m node transmission range. SEDUM-D-40 and SEDUM-F-40 use a 40m node transmission range. The experimental results follow SEDUM-D-40>SEDUM-D-20≈SEDUM-F-40>SEDUM-F-20. Since a larger transmission range enables a node to contact more nodes, SEDUM-F and SEDUM-D in the 40m node transmission range generate higher success rates than SEDUM-F and SEDUM-D with 20m transmission range. Since the duration utility can more accurately reflect the communication capacity between two nodes, the transmission success rate in SEDUM-D is much greater than in SEDUM-F with the same transmission range. A comparison of Figures 6.38 (a) and (b) shows that as the message size increases, the success rate of SEDUM-F decreases, while that of SEDUM-D remains almost identical. A larger message size increases the probability that messages are dropped during transmission due to broken links. This is consistent with Theorem 5.1.1, which implies that larger messages reduce throughput between two nodes. Nodes with a high frequency utility do not necessarily have a long communication time for each contact. A relay node may fail to send a complete message because of the

short communication period between two nodes despite their frequent meetings. The result confirms the higher accuracy of the duration utility than the frequency utility in reflecting the node transmission capacity.



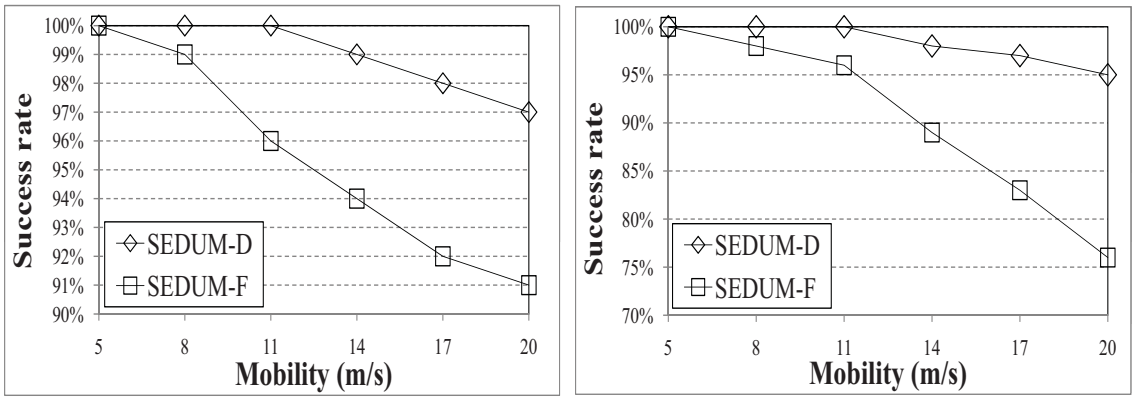
(a) Message size = 0.4Mb

(b) Message size = 4Mb

Figure 6.38: Comparison of duration utility and frequency utility based routings versus buffer size.

Figures 6.39 (a) and (b) show the relationship between the success rate and node mobility with 0.4Mb and 4Mb message sizes, respectively. As the node mobility increases, the message delivery throughput of both SEDUM-D and SEDUM-F decreases. As the node mobility increases, the expected contact duration between the nodes decreases, resulting in fewer messages transferred between nodes due to a limited contact time. We also observe a slight decrease in SEDUM-D and a dramatic decrease in SEDUM-F. This discrepancy is because the duration utility can more accurately reflect the communication capacity between nodes based on the node movement patterns. Using the duration utility, messages are always forwarded to nodes with higher communication capacities with the destination nodes, which leads to a higher transmission success rate. SEDUM-F uses contact frequency as the routing utility that helps route a message to a node that frequently meets the destination. As node mobility increases, the contact duration between nodes decreases even if they have high contact frequency, thereby increasing the probability of message drops. Consequently,

SEDUM-F decreases more quickly than SEDUM-D as node mobility increases.



(a) Message size = 0.4Mb (b) Message size = 4Mb

Figure 6.39: Comparison of duration utility and frequency utility based routings versus mobility.

Comparing Figures 6.39 (a) and (b), similar to Figures 6.38 (a) and (b), we notice that the performance of SEDUM-F is affected more by the message size than SEDUM-D for the same reason. The frequency utility mainly captures the familiar stranger attribute of the node movement patterns. A node with a high frequency utility to another node may have small contact duration with the node in a contact, which makes large-size messages more likely to be dropped. The duration utility can capture both colocation and familiar stranger attributes of node movement patterns. That is, the duration utility considers not only node contact duration but also contact frequency. Since nodes with long contact durations can transmit messages with large sizes, the transmission success rate of SEDUM-D is only marginally affected by the message size. These experimental results are in line with Theorem 5.1.2.

6.4 Summary

In this chapter, we evaluated the performance of the proposed mechanisms in terms of trustworthiness, scalability and efficiency. The evaluation results for trustworthiness show that compared to the reputation and price systems, the integrated system can provide the highest cooperation incentives. The integrated system is also more effective to detect the selfish nodes and sequentially reduce the system packet dropping rate of the reputation and price systems. The evaluation results for scalability show that ARM can provide efficient reputation management in MANETs with the lowest overhead compared to the reputation and price systems. Also, ARM can effectively identify falsified, conspiratorial and misreported information so as to provide accurate node reputations that truly reflect node behaviors. The evaluation results for routing efficiency show that the duration utility in SEDUM can achieve a higher packet transmission success rate compared to the frequency utility. SEDUM can also achieve the highest message delivery throughput and lowest message delivery delay compared to the spay and wait and epidemic routing algorithms.

Chapter 7

Conclusions and Future Work

7.1 Summary and Findings

As distributed wireless systems require all nodes in the network to cooperatively conduct a task, encouraging the cooperation of the nodes is crucial for the proper function of these systems. Reputation and price systems are two main approaches to deal with the cooperation problem in distributed wireless networks. However, the current reputation and price systems cannot effectively encourage the cooperation of the nodes in the system. Moreover, these two systems result in high overhead for reputation or price management as well as a low efficiency in selfish and colluding node detection. In this dissertation, we proposed mechanisms for distributed wireless systems to enhance trustworthiness, scalability, and efficiency.

To increase the trustworthiness of the distributed wireless systems, we initially analyzed the underlying cooperation incentives of the reputation and price systems and the defenseless system using cooperative and non-cooperative game models in Chapter 3. In

the cooperative game we found that each node earns the maximum benefit only when they form a grand coalition, in which all nodes in the system are cooperative. To form such a coalition, however, the cooperative strategy should be enforced by a third party. Based on the non-cooperative game model, we identified several problems that limit the cooperation incentives provided by both reputation and price systems. The strategies of using a threshold to determine the trustworthiness of a node in the reputation system and the strategies of rewarding cooperative nodes in the price system may be manipulated by clever but selfish nodes. These limitations will result in a high packet dropping rate of the whole system. Based on the investigation results, we have proposed an integrated system to leverage the advantages of both the reputation and price system, by integrating the misbehavior detection mechanism from the reputation system and the cooperation incentive mechanism from the price system. The integrated system can also overcome their individual disadvantages. The theoretical analysis and simulation results show that the integrated system can provide higher cooperation incentives than the reputation and price system alone in terms of higher payoffs for the cooperation strategy and make the cooperative strategy to be the NE and Pareto-optimal. The system is also more effective in selfish node detection, which can greatly reduce the packet dropping rate of the whole system.

To achieve high scalability, in Chapter 4, we further proposed a hierarchical Account-aided Reputation Management system (ARM) based on the theoretical results in Chapter 3, which can efficiently and effectively deter node selfish behaviors and provide cooperation incentives with small overhead. ARM builds an underlying locality-aware DHT infrastructure to efficiently collect global reputation information in the entire system for node reputation evaluation, which avoids periodical message exchange, reduces information redundancy, and more accurately reflects a node's trustworthiness. ARM has functions of reputation management and account management, the integration of which fosters the cooperation incentives and non-cooperation deterrence. ARM can detect the uncooperative nodes that gain fraud-

ulent benefits while still being considered as trustworthy in previous reputation and price systems. Also, it can effectively identify falsified, conspiratorial, and misreported information to provide accurate node reputations that truly reflect node behaviors. The simulation results show that ARM can greatly reduce the system overhead compared to the reputation and price systems alone as well as accurately calculate the global reputation of the nodes and effectively detect selfish and colluding nodes.

Based on the trustworthy and scalable mechanisms, we proposed a Social nEtwork and Duration Utility based distributed Multi-copy routing protocol (SEDUM) for DTNs in Chapter 5. In many DTN applications, the movements of mobile devices have certain patterns as the devices in these scenarios are the extensions of the hosts (i.e., human or animals), which normally exhibit colocation and familiar stranger features. We design a new metric, "duration utility", captures both colocation and familiar stranger attributes for message routing. Taking advantage of the duration utility, SEDUM fully exploits node movement patterns in the social network to increase delivery throughput and decrease delivery delay while generating low overhead. SEDUM replicates a new message to a certain number of nodes, which hold the replicas until meeting other nodes with higher duration utilities to the destinations. A message is forwarded in this way until one of its replicas reaches its destination. SEDUM includes a buffer management mechanism to improve performance. We also used a Markov chain to calculate the minimum number of copies of a message for a given delivery delay. The simulation results show that the proposed routing protocol outperforms the epidemic routing and spray and wait routing in terms of higher message delivery throughput, lower message delivery delay, lower message delivery overhead, and higher packet delivery success rate.

Chapter 6 presents the evaluation results for trustworthy, scalable and efficient distributed wireless system. The evaluation results for trustworthiness show that compared to the reputation and price systems, integrated system can provide highest cooperation

incentives. The Integrated system is also more effective to detect the selfish nodes and sequentially reduce the packet dropping rate of the whole system compared to the reputation and price systems. The evaluation results for scalability show that ARM can provide efficient reputation management in MANETs with lowest overhead compared to reputation and price systems. Also, ARM can effectively identify falsified, conspiratorial and misreported information so as to provide accurate node reputations that truly reflect node behaviors. The evaluation results for routing efficiency shows that duration utility can achieve a higher packet transmission success rate compared to the frequency utility. SEDUM can also achieve the highest message delivery throughput and lowest message delivery delay compared to spay and wait and epidemic routing algorithm.

7.2 Future Work

In the future, the following issues can be addressed.

Security in Communication. In the ARM system, the reputation managers are encouraged to manage the reputation and account values of all nodes in the system. However, such reputation managers are vulnerable to distributed deny of service (DDOS) attacks by malicious nodes. The malicious nodes can send many junk messages to reputation managers to prevent these managers from receiving reputation reporting from normal nodes. How to prevent the distributed system from DDOS attack is an interesting subject for future research.

Hybrid Networks. In this dissertation, we focused on the trustworthiness, scalability, and efficiency of distributed wireless systems such as MANETs and DTNs. The use of hybrid wireless networks is also a promising communication network in the future. A hybrid wireless network is a combination of a MANET and an infrastructure network. In a hybrid network, base stations in the infrastructure act as relays for mobile nodes in MANET for

long distance communications and Internet access, while MANET extends the coverage of the infrastructure network. Given such a new network model, how to ensure the trustworthiness, scalability, and efficiency of the communication between mobile nodes is also an interesting topic.

Altruism Analysis in Game Theory. In this dissertation, we used game theory models to evaluate the incentives provided by reputation systems, price systems, and the proposed integrate system. Game theory assumes that all the nodes in the game are self-interested. However, in reality, such an assumption does not always hold. Altruism is also common in our society. Therefore, how does altruism of the node affect the cooperation modeling for reputation and price systems will be an interesting topic for future research.

Real System Implementation. In this dissertation, theoretical modeling and simulation are used as major approaches to evaluate our proposed mechanisms. In the future, we would like to implement the mechanisms proposed in this dissertation in real systems such as mobile phone applications. By tracing interactions between real users through real applications, we can evaluate the effectiveness of our proposed mechanisms in the real-world scenario, which can provide more realistic indication to improve the proposed mechanisms in the dissertation.

Bibliography

[1] Next Generation Smartphones Players, Opportunities & Forecasts 2008-2013. Technical report, Juniper Research, 2009.

[2] Delay Tolerant Networking Research Group, June 2012. http://www.denrg.org.

[3] iPhone 4S Clocked at 800MHz, Still Crushes iPhone 4 (and Everyone Else) as Advertised, June 2012. http://9to5mac.com/2011/10/11/iphone-4s-geekbench-glbenchmark/.

[4] The Network Simulator NS-2, June 2012. http://www.isi.edu/nsnam/ns/.

[5] The State of the Smartphone Market, June 2012. http://www.allaboutsymbian.com/.

[6] A. Aram and C. Singh and S. Sarkar and A. Kumar. Cooperative Profit Sharing in Coalition Based Resource Allocation in Wireless Networks. In *Proc. of IEEE International Conference on Computer Communications*, Rio de Janeiro, Brazil, 19-25 April 2009.

[7] B. N. Levine A. Balasubramanian and A. Venkataramani. DTN Routing as a Resource Allocation Problem. In *Proc. of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Kyoto, Japan, August 27-31 2007.

[8] A. Duran and C. Shen. Mobile Ad Hoc P2P File Sharing. In *Proc. of IEEE Wireless Communications and Networking Conference*, Atlanta, Geogia, USA, March 21-25 2004.

[9] A. Lindgren and A. Doria and O. Schelen. Probabilistic Routing in Intermittently Connected Networks. *ACM Sigmobile Mobile Computing and Communication Review*, 7(3), 2003.

[10] E. Altman, A. A. Kherani, P. Michiardi, and R. Molva. Non-cooperative Forwarding in Ad-hoc Networks. *Lecture Notes in Computer Science*, 3462(10), 2005.

[11] T. Anantvalee and J. Wu. Reputation-based System for Encouraging the Cooperation Nodes in Mobile Ad Hoc Networks. In *Proc. of IEEE International Conference on Communications*, Glasgow, Scotland, June 24-28 2007.

[12] B. Pasztor and M. Musolesi and C. Mascolo. Opportunistic Mobile Sensor Data Collection with SCAR. In *Proc. of IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, Pisa, Italy, October, 8-11.

[13] A. Balasubramanian, B. Levine, and A. Venkataramani. DTN Routing as a Resource Allocation Problem. In *Proc. of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Kyoto, Japan, August 27-31 2007.

[14] S. Bansal and M. Baker. Observation-based Cooperation Enforcement in Ad Hoc Networks. *Networking and Internet architecture, http://arXiv.org/abs/cs/0307012*, 2003.

[15] S. P. Borgatti, A. Mehra, D. J. Brass, and G. Labianca. Network Analysis in the Social Sciences. *Science*, 323(5916), 2009.

[16] S. Buchegger and J. Y. Le Boudec. A Robust Reputation System for Mobile Ad Hoc Networks. In *Proc. of Workshop on economics of peer-to-peer systems*, Harvard University, June 4-5 2004.

[17] J. Burgess, B. Gallagher, D. Jensen, and B. Levine. MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In *Proc. of IEEE International Conference on Computer Communications*, Barcelona, Spain, April 23 - April 29 2006.

[18] B. Burns, O. Brock, and B. N. Levine. MV Routing and Capacity Building in Disruption Tolerant Networks. In *Proc. of IEEE International Conference on Computer Communications*, Miami, USA, March 13-17 2005.

[19] C. Saraydar and N. Mandayam and D. Goodman. Efficient Power Control Via Pricing in Wireless Data Networks. *IEEE Transactions on Communications*, 50(2), 2002.

[20] C. Z. Mooney. *Monte Carlo Simulation: Quantitative Applications in The Social Sciences*. Sage Publications Inc, ISBN-10: 0803959435, 1997.

[21] J. Cai and U. Pooch. Allocate Fair Payoff for Cooperation in Wireless Ad Hoc Networks Using Shapley Value. In *Proc. of IEEE International Parallel and Distributed Processing Symposium*, Santa Fe, New Mexico, USA, 26-30 April 2004.

[22] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of Human Mobility on the Design of Opportunistic Forwarding Algorithms. In *Proc. of IEEE Conference on Computer Communications*, Barcelona, Spain, April 23-29 2006.

[23] L. J. Chen, C. H. Yu, T. Sun, Y. C. Chen, and H. H. Chu. A Hybrid Routing Approach for Opportunistic Network. In *Proc. of ACM Workshop on Challenged Networks*, Pisa, Italy, Septerber 15 2006.

[24] S. Chen, J. Zhang, and Q. Gao. An Efficient Hybrid Routing based on Contact History in Delay Tolerant Networks. In *Proc. of International Conference On Wireless And Optical Communications Networks*, Indore, India, September 6-8 2010.

[25] V. Conan, J. Leguay, and T. Friedman. Fixed Point Opportunistic Routing in Delay Tolerant Networks. *IEEE Journal on Selected Areas in Communications*, 26(5), 2008.

[26] T. H. Cormen. *Introduction to Algorithms*. The MIT press, ISBN-10: 0262033844, 2001.

[27] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco. Socially-aware Routing for Publish-subscribe in Delay-tolerant Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, (5), 2008.

[28] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco. Socially-aware Routing for Publish-subscribe in Delay-tolerant Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 26(5), 2008.

[29] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. *Springer Mobile Computing* , 353(10), 1996.

[30] E. M. Daly and M. Haahr. Social Network Analysis for Routing in Disconnected Delay-tolerant MANETs. In *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Montreal, Canada, September 9-14 2007.

[31] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages. In *Proc. of ACM international Symposium on Mobile Ad hoc Networking and Computing*, Annapolis, MD, USA, June 1-3 2003.

[32] E. N. Barron. *Game Theory - An Introduction*. The Wiley Bicentennial, ISBN: 978-0-470-17132-5, 2008.

[33] G. A. Dirac. Some Theorems On Abstract Graphs. *London Math*, 3-2(1), 1952.

[34] W. Gao, Q. Li, B. Zhao, and G. Cao. Multicasting in Delay Tolerant Networks: a Social Network Perspective. In *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing*, New Orleans, USA, May 18-21 2009.

[35] J. Ghosh, S. J. Philip, and C. Qiao. Sociological Orbit Aware Location Approximation and Routing (SOLAR) in MANET. *Ad Hoc Networks*, 5(2), 2007.

[36] H. Gharavi. Multichannel Mobile Ad Hoc Links for Multimedia Communications. *Proceeding of The IEEE*, 96(1).

[37] H. Janzadeh and K. Fayazbakhsh and M. Dehghan and M. S. Fallah. A Secure Credit-Based Cooperation Stimulating Mechanism for MANETs Using Hash Chains. 25(8), 2009.

[38] H. Shen and Z. Li. Arm: An Account-Based Hierarchical Reputation Management System for Wireless Ad Hoc Networks. In *Proc. of International Workshop on Wireless Security and Privacy (WiSP'08), held in conjunction with The 28th International Conference on Distributed Computing Systems*, Beijing, China, June 17 2008.

[39] H. Wu and R. Fujimoto and R. Guensler and M. Hunter. MDDV: Mobility-Centric Data Dissemination Algorithm for Vehicular Networks. In *Proc. of ACM Workshop on Vehicular Ad Hoc Networks*, Philadelphia, PA, USA, October 1 2004.

[40] S. S. Haykin. *Kalman filtering and neural networks.* Wiley-Interscience, ISBN: 978-0-471-36998-1, 2001.

[41] P. Hui and J. Crowcroft. How Small Labels Create Big Improvements. In *Proc. of ACM International Conference on Emerging Networking EXperiments and Technologies*, New York, NY, USA, 10-13 December 2007.

[42] P. Hui, J. Crowcroft, and E. Yoneki. Bubble Rap: Social-based Forwarding in Delay Tolerant Networks. *IEEE Transactions on Mobile Computing*, 10(11), 2010.

[43] J. Crocraft and R. Gibbens and F. Kelly and S. Ostring. Modeling Incentives for Collaboration In Mobile Ad Hoc Networks. *Performance Evaluation*, 57(4), 2004.

[44] J. E. Suris and L. A. Dasilva and Z. Han and A. B. Mackenzie. Cooperative Game Theory for Distributed Spectrum Sharing. In *Proc. of International Conference on Communications*, Glasgow, Scotland, June 24-28 2007.

[45] J. J. Jaramillo and R. Srikant. Darwin: Distributed and Adaptive Reputation Mechanism for Wireless Networks. In *Proc. of International Conference on Mobile Computing and Networking*, Montreal, Canada, September 9-14 2007.

[46] J. Widmer and J. Y. L. Boudec. Network Coding for Efficient Communication in Extreme Networks. In *Proc. of ACM Workshop on Delay Tolerant Networking and Related Topics* , Philadelphia, PA, August 26.

[47] S. Jain, K. Fall, and R. Patra. Routing in a Delay Tolerant Network. In *Proc. of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Portland, Oregon, USA.

[48] A. Jardosh, E. M. Belding, K. C. Almeroth, and S. Suri. Towards Realistic Mobility Models for Mobile Ad Hoc Networks. In *Proc. of International Conference on Mobile Computing and Networking*, San Diego, USA, September 14-19 2003.

[49] K. Balakrishnan and J. Deng and V. K. Varshney. TWOACK: Preventing Selfishness in Mobile Ad Hoc Netwotks. In *Proc. of IEEE Wireless Communications and Networking Conference*, New Orleans, LA, March 13-17 2005.

[50] K. Liu and J. Deng and P. K. Varshney and K. Balakrishnan. An Acknowledgment-Based Approach for The Detection of Routing Misbehavior in MANETs. *IEEE Transactions on Mobile Computing*, 6(5), 2007.

[51] H. Kameda and E. Altman. Inefficient Noncooperation in Networking Games of Common-pool Resources. *IEEE Journal on Selected Areas in Communications*, 26(7), 2008.

[52] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and Panigrahy R. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. In *Proc. of ACM Symposium on the Theory of Computing*, El Paso, Texas, USA, May 4-6 1997.

[53] A. Keranen, J. Ott, and T. Karkkainen. The ONE Simulator for DTN Protocol Evaluation. In *Proc. of International Conference on Simulation Tools and Techniques*, Rome, Italy, March 2-6 2009.

[54] L. Anderegg and S. Eidenbenz. Ad Hoc-VCG: A Truthful And Cost-Efficient Routing Protocol for Mobile Ad Hoc Networks With Selfish Agents. In *Proc. of ACM International Conference on Mobile Computing and Networking*, San Diego, CA, USA, September 14-19 2003.

[55] L. B. Koralov and Y. G. Sinai. *Theory of Probability and Random Processes*. Springer, ISBN-10: 3540254846, 2007.

[56] L. Buttyan and J. Hubaux. Enforcing Service Availability in Mobile Ad-Hoc Wans. In *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Boston, MA, USA, August 11 2000.

[57] L. Buttyan and J. P. Hubaux. Stimulating Cooperation in Self-Organzing Mobile Ad Hoc Network. *ACM Journal for Mobile Networks and Applications*, 8(5), 2003.

[58] L. Cao and H. Zheng. Distributed Spectrum Allocation via Local Bargaining. In *Proc. of IEEE Conference On Sensor and Ad Hoc Communications and Networks*, Santa Clara, California, USA, 26-29 September 2005.

[59] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong. SLAW: A Mobility Model for Human Walks. In *Proc. of IEEE International Conference on Computer Communications*, Rio de Janeiro, Brazil, April 19 - April 25 2009.

[60] F. Li and J. Wu. MOPS: Providing Content-based Service in Disruption Tolerant Networks. In *Proc. of International Conference on Distributed Computing Systems*, Montreal, Quebec, Canada, June 22-26 2009.

[61] Z. Li and H. Shen. Game-theoretic analysis of cooperation incentive strategies in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 11(8):1287–1303, 2011.

[62] Z. Li and H. Shen. SEDUM: Exploiting Social Networks in Utility-Based Distributed Routing for DTNs. *IEEE Transactions on Computers*, PP(99):1, accepted to appear.

[63] J. Luo, X. Liu, and M. Fan. A Trust Model Based on Fuzzy Recommendation for Mobile Ad-hoc Networks. *Computer Network*, 53(14), 2009.

[64] M. Felegyhazi and L. Buttyan and J. P. Hubaux. Nash Equilibria of Packet Forwarding Strategies in Wireless Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 5(5), 2006.

[65] M. J. Neely and E. Modiano. Improving Delay in Ad-Hoc Mobile Networks Via Redundant Packet Transfers. In *Proc. of the Conference on Information Sciences and Systems*, Johns Hopkins University, March 2003.

[66] M. Jakobsson and J. Hubaux and L. Buttyan. A Micropayment Scheme Encouraging Collaboration in Multi-hop Cellular Networks. In *Proc. of International Conference on Financial Cryptograph*, Gosier, Guadeloupe, January 27-30 2003.

[67] M. T. Refaei and L. A. DaSilva and M. Eltoweissy and T. Nadeem. Adaptation of Reputation Management Sytems to Dynamic Network Conditions in Ad Hoc Networks. *IEEE Transaction On Computers*, 59(5), 2010.

[68] Marti, S. and Giuli, T. J. and Lai, K. and Baker, M. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Proc. of International Conference on Mobile Computing and Networking*, Boston, MA, August 6-11 2000.

[69] Marti, S., and Giuli, T. J. and Lai, K. and Baker, M. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Proc. of ACM International Conference on Mobile Computing and Networking*, Boston, MA, USA, August 6-11 2000.

[70] J. Mundinger and J. Le Boudec. Analysis of a reputation system for mobile ad-hoc networks with liars. *Performance Evaluation*, 65(3-4), 2008.

[71] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive Routing for Intermittently Connected Mobile Ad Hoc Networks. In *Proc. of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Taormina Giardini Naxos, June 13-16 2005.

[72] N. Bansal and Z. Liu. Capacity, Delay and Mobility in Wireless Ad-Hoc Networks. In *Proc. of IEEE International Conference on Computer Communications*, San Franciso, CA, USA, March 30 - April 3 2003.

[73] P. D. Straffin. *Game Theory and Strategy*. The Mathematical Association of Amecica, ISBN-10: 0883856379, 1993.

[74] P. Dewan and P. Dasgupta and A. Bhattacharys. On Using Reputations in Ad Hoc Networks To Counter Malicious Nodes. In *Proc. of International Conference on Parallel and Distributed Systems*, Newport Beach, CA, USA, July 7-9 2004.

[75] P. Juang and H. Oki and Y. Wang, M. Martonosi and L. S. Peh, and D. Rubenstein. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences With Zebranet. In *Proc. of International Conference on Architectural Support for Programming Languages and Operating Systems* , San Jose, California, USA, October 5-9 2002.

[76] P. Michiardi and R. Molva. CORE: A Collaborative Reputation Mechanism To Enforce Node Cooperation in Mobile Ad Hoc Networks. In *Proc. of the IFIP Conference on Communications and Multimedia Security*, Portoroz, Slovenia, September 26-27 2002.

[77] E. Paulos and E. Goodman. The Familiar Stranger: Anxiety, Comfort, and Play in Public Places. In *Proc. of International Conference for Human-computer Interaction*, Vienna, Austria, April 24-29 2004.

[78] Q. He and D. Wu and P. Khosla. SORI: A Secure and Objective Reputation-based Incentive Scheme for Ad-hoc Networks. In *Proc. of EEE Wireless Communications and Networking Conference* , Atlanta, Geogia, March 21-25 2004.

[79] Q. Li and D. Rus. Communication in Disconnected Ad Hoc Networks Using Message Relay. *ACM/IEEE Transactions On Networking*, 63(1), 2003.

[80] R. Etkin and A. Parekh and D. Tse. Spectrum Sharing for Unlicensed Bands. In *Proc. of IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks*, Baltimore, MD, USA, November 8-11 2005.

[81] S. Buchegger and J. Y. L. Boudec. A Robust Reputation System for Ad Hoc Networks. Technical report, cole Polytechnique Fdrale de Lausanne , 2003.

[82] S. Buchegger and J.-Y. L. Boudec. Performance Analysis of The Confidant Protocol. In *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Anapolis, MD, June 1-3 2003.

[83] S. Buchegger and J. Y. Leboudec. The Effect of Rumor Spreading in Reputation Systems for Mobile Ad Hoc Networks. In *Proc. of Workshop on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, INRIA Sophia-Antipolis, France, March 3-5 2003.

[84] S. Goel and T. Imielinski and K. Ozbay. Ascertaining Viability of Wifi Based Vehicle-To-Vehicle Network for Traffic Information Dissemination. In *Proc. of IEEE International Conference on Intelligent Transportation Systems*, Washington D.C, USA, October 3-6 2004.

[85] S. Zhong and J. Chen and Y. R. Yang. Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad Hoc Networks. In *Proc. of IEEE International Conference on Computer Communications*, San Francisco, USA, March 30 - April 3 2003.

[86] T. Small and Z. Haas. Resource and Performence Tradeoffs in Delay-tolerant Wireless Networks. In *Proc. of ACM Workshop on Delay Tolerant Networking and Related Topics*, Philadelphia, PA, August 26 2005.

[87] T. Spyropoulos, K. Psounis, and C. Raghavendra. Efficient Routing in Intermittently Connected Mobile Networks: The Single-copy Case. *ACM/IEEE Transactions on Networking*, 16(1), 2007.

[88] Stoica, I. and Morris, R. and et al. Chord: A Scalable Peer-to-peer Lookup Protocol For Internet Applications. *IEEE/ACM Transactions on Networking*, 11(1), 2003.

[89] T. Moscibroda and S. Schmid. On Mechanism Design Without Payments for Throughput Maximization. In *Proc. of IEEE International Conference on Computer Communications*, Rio de Janeiro, Brazil, April 19-25 2009.

[90] T. Spyropoulos and K. Psounis and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. of ACM SIGCOMM workshop on Delay-tolerant networking* , New York, NY, USA, August 22 2005.

[91] A. Urpi, M. Bonuccelli, and S. Giordano. Modeling Cooperation in Mobile Ad Hoc Networks: a Formal Description of Selfishness. In *Proc. of Workshop on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, INRIA Sophia-Antipolis, France, March 3-5 2003.

[92] V. Srinivasan. Cooperation in Wireless Ad Hoc Networks. In *Proc. of IEEE International Conference on Computer Communications*, San Franciso, CA, USA, March 30 - April 3 2003.

[93] V. Srivastava and J. Neel and A. B. Mackenzie and R. Menon and L. A. Dasilva and J. E. Hicks and J. H. Reed and R. P. Gilles. Using Game Theory To Analyze Wireless Ad Hoc Networks. *IEEE Communications Surveys and Tutorials*, 7(4), 2005.

[94] A. Vahdat and D. Becker. Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-200006, Duke University, 2000.

[95] W. Saad and Z. Han and M. Debbah and A. Hjorungnes and T. Basar. Coalitional Games for Distributed Collaborative Spectrum Sensing in Cognitive Radio Networks. In *Proc. of IEEE International Conference on Computer Communications*, Rio de Janeiro, Brazil, 19-25 April 2009.

[96] Y. Wang and H. Wu. Delay/Fault-Tolerant Mobile Sensor Network (DFT-MSN): A New Paradigm for Pervasive Information Gathering. *IEEE Transactions on mobile computing*, 6(9), 2006.

[97] Y. Wang and S. Jain and M. Martonosi and K. Fall. Erasure-Coding Based Routing for Opportunistic Networks. In *Proc. of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Philadelphia, PA, USA , August 22-26 2005.

[98] L. Ying, S. Yang, and R. Srikant. Optimal Delay-Throughput Tradeoffs in Mobile Ad Hoc Networks. *IEEE Transactions on Information Theory*, 54(9), 2008.

[99] Z. Han and K. Liu. Noncooperative Power-Control Game and Throughput Game Over Wireless Networks. *IEEE Journal on Selected Areas In Communications*, 25(6), 2005.

[100] Z. Han and Z. Ji and K. Liu. Fair Multiuser Channel Allocation for Ofdma Networks Using Nash Bargaining Solutions and Coalitions. *IEEE Transaction On Communication*, 26(5), 2005.

[101] Z. Li and H. Shen. Analysis of The Cooperation Strategies in Mobile Ad Hoc Networks. In *Proc. of International Workshop on Wireless and Sensor Networks Security*, Atlanta, USA, September 29 2008.

[102] Z. Li and H. Shen. Utility-based Distributed Routing in Intermittently Connceted Networks. In *Proc. of International Conference on Parallel Processing*, Portland, Oregon , USA, September 8-12 2008.

[103] Z. Li and H. Shen. Analysis of a Hybrid Reputation Management System for Mobile Ad Hoc Networks. In *Proc. of International Conference on Computer Communications and Networks*, San Francisco , USA, August. 2-6 2009.

[104] Z. Li and H. Shen. A Hierarchical Account-aided Reputation Management System for Large-Scale MANETs. In *Proc. of IEEE International Conference on Computer Communications*, Shanghai , China, April. 10-15 2011.

[105] X. Zhang, G. Neglia, J. Kurose, and D. Towsley. Performance Modeling of Epidemic Routing. *The International Journal of Computer and Telecommunications Networking*, 51(10), 2007.

[106] W. Zhao, M. Ammar, and E. Zegura. A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks. In *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Tokyo, Japan, May 24-26 2004.

[107] Zong, B. and Xu, F. and Jiao, J. and Lv, J. A Broker-Assisting Trust and Reputation System Based on Artificial Neural Network. In *Proc. of IEEE International Conference on Systems, Man and Cybernetics*, Nanjing, China, Oct. 11-14 2009.