INCENTIVE COMPATIBLE PROTOCOLS IN WIRELESS NETWORKS USING NETWORK CODING AND IN COGNITIVE RADIO NETWORKS USING COOPERATIVE RELAY

by

Haifan Yao

Sep 2013

A dissertation submitted to the
Faculty of the Graduate School of
the University at Buffalo, State University of New York
in partial fulfilment of the requirements for the
degree of

Doctor of Philosophy

Department of Computer Science and Engineering

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

ii

The thesis of Haifan Yao was reviewed by the following:

Sheng Zhong
Research Professor of Computer Science and Engineering
Thesis Advisor, Chair of Committee

Shambhu J. Upadhyaya
Professor of Computer Science and Engineering
Committee Member

Chunming Qiao
Professor of Computer Science and Engineering
Committee Member

Acknowledgments

First of all, I would like to wholeheartedly thank my dissertation advisor, Prof. Sheng Zhong for his professional supervision and thoughtfulness during my Ph.D. study. His valuable and constructive guidance always lead me towards the right direction in research. I am deeply grateful to my dissertation committee members, Prof. Chunming Qiao and Prof. Shambhu Upadhyaya, who enlightened me a lot and help me in finalizing this thesis. I thank my coauthors, Dr. Jiqiang Liu, Dr. Zhuo Hao, very much for their contributions in our collaborations. Thank you to my colleagues in our research group, Dr. Tingting Chen, Yuan Zhang and Yu Li. Their encouragement and support always give me power. Last but not least, I would thank my parents Xiaojun Zhang and Dong Yao, my wife Yi Shang and my beloved son Frank S. Yao. They are the reason why I am here and will keep going.

Table of Contents

List of Tables

List of Figures

Abstract

In this thesis, several studies of incentive compatibility in wireless networks were presented. In particular, the thesis focuses on two subareas of wireless networks: network coding technique and cooperative relay service in cognitive radio networks. In aspect of network coding, I presented a general cheating detection scheme in which a node can initiate a threshold decision session when it believes its payment was miscalculated. Then I extended the work to the incentive scheme for packet forwarding and payment reduction in wireless networks using XOR network coding. In aspect of cooperative relay, I designed a scheme that provides incentives for the secondary users to truthfully report information to the primary user, which is the first cheat proof scheme for cooperative relay in cognitive radio networks. In this work, we apply solution concepts in game theory to rigorously guarantee that our schemes will stimulate users to cooperate to the best of their interests. Then algorithms for cooperative relay protocols with general utility functions were presented, since we find there is no existing protocol that provides Nash equilibrium solutions for general cases. Lastly, I designed algorithms for cooperative relay among secondary users, which guarantees that, under a precondition called No Monopoly, all relay nodes have incentives to truthfully share their relay information.

Chapter 1

Introduction and Technical Preliminaries

1.1 Introduction

In this thesis, I present several studies of incentive compatibility, which I have completed during my PhD research. In particular, I have focused on two sub-areas of wireless networks, namely network coding and cooperative relay in cognitive radio networks.

Wireless mesh networks have been widely deployed to provide broadband network access to schools, communities, and participants of various events. It is very challenging and highly important to improve the performance of wireless mesh networks so that the throughput scalability of such networks can meet the

needs of different users. One way to achieve significantly better performance for wireless mesh networks is to apply a technique called network coding [52]. Unlike in conventional networks, in wireless networks using network coding, intermediate nodes do not store and forward the same packets as sent by the source node. In stead, intermediate nodes forward new coded packets computed by themselves from the packets they have received [56, 57, 58, 59, 60]. Hence, the data is actually mixed at each intermediate node before it is forwarded. This idea of mixing data at intermediate nodes takes advantage of the broadcast nature of wireless transmissions, and achieves great performance gains.

Many wireless mesh networks have user contributed wireless devices as their nodes. Since users normally have their own interests, economic incentives become a crucial problem. A selfish or economically rational user may let her wireless device deviate from the communication protocol, as long as the deviation is beneficial to herself. However, this deviation may harm the network's performance, or even lead the network to stop functioning in the worst case. Therefore, a lot of work has been done so far using payment based scheme to make the communication protocol incentive compatible, so that nodes have incentives to faithfully follow the protocol.

In a payment based system, a source node (or, sometimes, a destination node) is required to pay a certain amount of virtual money to intermediate nodes who transmit its data. Correspondingly, nodes that help transmitting other nodes' data can earn virtual money. Such a scheme will efficiently save the network resources and improve the efficiency of data transmission, since the source node can not transmit its data unlimitedly and intermediate nodes are

pleased to provide their communication resources for forwarding other nodes' messages. In addition, under well designed payment schemes, a node may find that being selfish will hurt its own benefit and thus have incentives to be honest.

Nevertheless in reality, the node in charge of payment calculation may be selfish or corrupted. It may miscalculate the payment for its own good or collude with others for the benefit of a cheating group. For example, if the source is responsible for calculating the payment for each intermediate nodes (which is quite common as the source collects routing information in many routing protocols), it may decrease the payment deliberately to save its own virtual money. Or in another case, if the payment is calculated by an intermediate node, the payment due to this intermediate node is calculated higher than it should be in order to benefit the intermediate node itself. This can lead to serious problems especially when there is no authority entity available for real time payment inspection. The entire system may collapse because of these selfish behaviors. Therefore we need a cheating detection scheme to fight against the misbehavior of payment calculator.

We notice that in case a central authority is absent, there is no work that prevents payment cheating for existing payment schemes. Thus we present a general cheating detection scheme in which a node can initiate a threshold decision session when it considers its payment miscalculated. Then we extend our work to the incentive scheme for packet forwarding and payment reduction in wireless networks using XOR network coding.

On the other hand, cognitive radio networks [62][63][64] have received a lot of attention in recent years, because they allow secondary users to detect the spectrum not used by primary users and thus improves the utilization of spec-

trum. In cognitive radio networks, *cooperatively relay* [65]-[71] is a new approach in which nodes help each other to relay traffic, so that better performance can be achieved.

Two types of cooperative relay have been proposed in cognitive radio networks these years. The first type of cooperative relay is between primary and secondary users, where secondary users seek opportunities to relay data for the primary user in exchange for some time for its own data transmission in primary users free spectrum. This type of cooperative relay [65][66],[81]-[89] has been extensively studied recently; lots of interesting results have been obtained, including some based on game theory. Although this kind of relay is a promising method to achieve better efficiency of the spectrum resource, works in this category generally assume the primary user is a licensed user who owns the spectrum property and may choose to lease portions of the spectrum to the secondary users. Thus they might not be suitable for other spectrum sharing models.

The second kind of cooperative relay is proposed to improve the spectrum utilization when primary users are not necessarily involved. In particular, Jia, Zhang and Zhang [67][68] notice that the spectrum availability of secondary users in cognitive radio networks is heterogeneous, and secondary users may have different traffic demands. Consequently, they propose protocols in which secondary users use their spare spectrum to relay traffic for other secondary users, in order to improve the spectrum usage of all secondary users. Thereafter, many studies have been conducted on this type of cooperative relay [69][70][71].

In the second half of this thesis, we first design a scheme that provides incentives for the secondary users to truthfully report information to the primary

user, which is the first cheat proof scheme for cooperative relay in cognitive radio networks. In this work, we apply solution concepts in game theory to rigorously guarantee that our schemes will stimulate users to cooperate to the best of their interests. Then we present an algorithms for cooperative relay protocols with general utility functions. Because we find there is no existing protocol that provides NE solutions for general cases. Finally, we present another incentive compatible protocol for the second type of cooperative relay as we mentioned above, so that secondary users will never deviate from designed cooperative relay service even if they are selfish.

The rest of this thesis is organized as follows: We review several game theoretic concepts and network protocols that are important to our work in the rest of Chapter 1. Two work is presented in Chapter 2 that focus on providing security and incentive compatibility for protocols in network coding. In Chapter 3, we consider both two types of cooperative relay and aim to propose effective schemes that suppress cheating during service.

1.2 A Review of Some Concepts in Game Theory

1.2.1 Non-cooperative Strategic Game

In a non-cooperative strategic game, the set of players is denoted by N. Any player $i \in N$ tries to maximize its own utility, denoted by u_i. A strategy profile s is a set of strategies which specifies actions of all players in a game. Denote by s_i the strategy player $i \in N$ chooses in strategy profile s, and by s_{-i} the strategies in profile s chosen by all players other than player i. Note: it is a convention in game theory that subscript $-i$ represents all players other than i. $u_i(s_i, s_{-i})$

denotes the utility of user i while all users choose actions in strategy profile s.

In addition, we define two more important solution concepts: Nash equilibrium (NE) and dominant strategy equilibrium (DSE).

Definition 1. *A Nash equilibrium (NE) of a strategic game is a strategy profile s^*, such that for any $i \in N$ and for any possible strategy profile s, we have*

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*). \tag{1.1}$$

In other words, any player cannot benefit from unilateral deviation from the equilibrium.

Definition 2. *A dominant strategy equilibrium (DSE) of a strategic game is a strategy profile s^* such that for any $i \in N$ and for any possible strategy profile s,*

$$u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i}). \tag{1.2}$$

In other words, any player's strategy in the equilibrium is always the best possible.

Note that DSE is *stronger* than NE in the sense that it provides players with stronger incentives to stay at the equilibrium.

1.2.2 Extensive Game with Perfect Information

Definition 3. *An extensive game with perfect information consists of:*

1. *A set of players: N.*
2. *A set Λ of history sequences tracking player actions within two stages.*
3. *A number of action sets A_i for each player $i \in N$.*

4. *A function P that assigns some players for next move given a history $\lambda \in \Lambda$. Each player $i \in P(\lambda)$ is a player who has to take a move after history $\lambda \in \Lambda$. If $P(\lambda) = \emptyset$, then λ is called a terminal history, otherwise a non-terminal history.*

5. *A number of utility functions u_i of each player $i \in N$.*

Definition 4. *The subgame of the extensive game with perfect information $\Gamma = (N, \Lambda, P)$ that follows history λ is the extensive game $\Gamma(\lambda) = (N, \Lambda|_\lambda, P|_\lambda)$, where $\Lambda|_\lambda$ is the set of history sequence λ' where as $(\lambda, \lambda') \in \Lambda$, and $P|_\lambda(\lambda') = P(\lambda, \lambda')$.*

Definition 5. *The subgame perfect Nash equilibrium (SPNE) of an extensive game with perfect information $\Gamma = (N, \Lambda, P)$ is a strategy profile s^*, such that $\forall \lambda \in \Lambda, \forall i \in P(\lambda)$ and $\forall s_i|_\lambda$ in subgame $\Gamma(\lambda)$, we have*

$$u_i(s_i^*|_\lambda, s_{-i}^*|_\lambda) \geq u(s_i|_\lambda, s_{-i}^*|_\lambda). \tag{1.3}$$

In other words, a SPNE induces a NE in each subgame.

1.3 State of the Art

1.3.1 Network Coding Protocols in Wireless Networks

1.3.1.1 COPE

COPE [59] is an elegant XOR network coding protocol in wireless networks. Each node in COPE is equipped with a omni-directional antennae and set in promiscuous mode, so that they can snoop on all communications over the wireless medium.

In this network, source node has to compute its shortest path to the destination using a link-state routing protocol before transmitting its packets. This routing protocol should guarantee that each node has good knowledge of delivery probability between every pair of nodes. A routing protocol which uses the ETX metric [18] can be used, such that the delivery probabilities are periodically computed and broadcasted to all nodes in the network.

During data transmission, an intermediate node in one session may server as an intermediate node for other sessions. Packets from different sessions may be XOR-ed by this intermediate node, while ensuring each receiver on next hop has enough information to decode the native packet. In COPE, an intermediate node should aim to XOR as many packets as possible, in order to maximize the number of native packets delivered in a single transmission, which is called "opportunistic coding" [59]. COPE exploits the nature of wireless network and can nicely improve the system throughput.

1.3.1.2 MORE

Suppose there is a wireless network with a set V of nodes. For $i, j \in V$, $(i, j) \in E$ is the link from node i to node j. In MORE protocol [17], before each transmission session, the routing decision should be made, and the path from the source to the destination should be determined. In wireless networks using network coding, routing decision is made based upon the loss probability from each node on the path to its neighbors. Denote by $\epsilon_{i,j}$ the loss probability of link (i, j). We assume each node can attain its loss probability to each of its neighbors by periodically sending probing signals, and share this loss probability matrix with other nodes in the network [18]. Therefore each node will obtain a matrix

M_ϵ in which $m_{ij} = \epsilon_{i,j}$. There is a function $f()$ which computes the number of transmissions node i should make for each packet [17]:

$$z_i = f(S, D, i, \{(i, j, \epsilon_{i,j}) | (i, j) \in E\}),$$

where S and D are the source and destination respectively.

1.3.2 Cooperative Relay Protocols in Cognitive Radio Networks

Suppose there is a cognitive radio network, in which there are a primary sender PS and a primary receiver PR. The ordered pair (PS, PR) is called *the primary user*. A set of N secondary transmission pairs SU are working in the same spectrum band. This set $SU = \{(SS_i, SR_i) | i = 1, \ldots, N\}$, where each secondary sender SS_i always seeks opportunities to transmit data to its paired receiver SR_i. For simplicity, suppose that each pair (SS_i, SR_i) in SU represents a distinct secondary user i. Cooperative relay [68] is introduced by Zhang, et al., in this network in order to increase the system throughput as follows: PS distributes its data to (a subset of) secondary users, which relay the data to PR. In return, these secondary users who provide relay service can access channels under the primary user's permission. They assume all data relays involve only secondary senders, not secondary receivers. Hence, when we talk about data relays, we refer to "secondary users" and "secondary senders" interchangeably.

In [81], Simeone, et al. [81], show that cooperative relay can be used to improve the primary user's achievable transmission rate. They assume the channels are modeled as independent Gaussian random variables; all of them are assumed to be slow fading channels. They consider data transmissions in time slots; the channel variables are varying over all slots but are assumed to be con-

stants within each slot. Let P_0 be the power level used by the primary user, and P_i be the power level used by secondary user i. Denote by h_0 the complex channel gain between PS and PR, by h_{0i} the channel gain between PS and SS_i, by h_{i0} the channel gain between SS_i and PR, and by h_i the channel gain between SS_i and SR_i. They propose that secondary users can relay data transmission for the primary user and in return the primary user can allocate portions of her spectrum for secondary users. They also present a model of Stackelberg game for this problem. In this model, they present a nice proof for the existence of NE, as well as a necessary condition for the NE to be unique.

In [82], J. Zhang and Q. Zhang study the same problem but use a different model. Their model is also a Stackelberg game but is defined differently from Simeone, et al.'s. They calculate the utilities of users using the achieved transmission rates and the payments. In their model , they elegantly prove that following their protocol is a unique NE.

For cooperative relay, as in [82], the primary user chooses a subset S of secondary users based on the values of $|h_{i0}|$ and $|h_{0i}|$ (but *independent* from the values of $|h_i|$). The primary user also determines two parameters α and β ($0 \leq \alpha \leq 1, 0 \leq \beta \leq 1$). Accordingly, each time slot is divided into three phases: the first phase occupies $\alpha\beta$ of the slot, and is used for data distribution from PS to secondary users in the set S; the second phase occupies $\alpha(1-\beta)$ of the slot, and is used for PR to receive data from PS and those selected secondary users; the third phase is the remaining $(1-\alpha)$ of the slot. The secondary users in S can use the third phase for their own data transmissions.

In addition to providing relay service, each secondary user $i \in S$ also needs to make a payment c_i to the primary user for their use of the band, where c_i is

determined by secondary user i himself. The access time in the third phase is assigned to these secondary users based on the amounts of their payments. The method of this assignment depends on the protocol used. Regardless of which protocol is used, the transmission rates in the cooperative relay system can be calculated as follows:

$$\begin{cases} R_{PS}(S) = \log_2(1 + \frac{\min_{i \in S} |h_{0i}|^2 P_0}{N_0}), \\ R_{SP}(S) = \log_2(1 + \frac{|h_0|^2 P_0}{N_0} + \sum_{i \in S} \frac{|h_{i0}|^2 P_i}{N_0}), \\ R_i = \log_2(1 + \frac{|h_i|^2 P_i}{N_0}), \end{cases} \tag{1.4}$$

where R_{PS} is the transmission rate from the primary user to the secondary users in the first phase, R_{SP} is the transmission rate from the secondary users to the primary user in the second phase, and R_i is the transmission rate of secondary user (SS_i, SR_i) in the third phase. In the first equation of (1.4), $\frac{\min_{i \in S} |h_{0i}|^2 P_0}{N_0}$ represents the lowest received SNR from PS to all SUs. In the second equation of (1.4), $\frac{|h_0|^2 P_0}{N_0}$ represents the received SNR from PS to PR, and $\frac{|h_{i0}|^2 P_i}{N_0}$ represents the received SNR from secondary user i to PR. In the third equation of (1.4), $\frac{|h_i|^2 P_i}{N_0}$ represents the received SNR from the sender of i to the receiver of i. The transmission rate from PS to PR via cooperative relay is:

$$R_P(\alpha, \beta, S) = \min\{\alpha\beta R_{PS}(S), \alpha(1-\beta) R_{SP}(S)\}. \tag{1.5}$$

Intuitively, $\alpha\beta R_{PS}(S)$ is the rate from PS to the relay nodes, and $\alpha(1-\beta) R_{SP}(S)$ is the rate from the relay nodes to PR. Hence, the smaller of these two rates is the effective rate from PS of PR.

Chapter 2

Incentive Compatible Protocols in Wireless Networks using Network Coding

2.1 Introduction

Recently, wireless mesh networks have been widely deployed to provide broadband network access to schools, communities, and participants of various events. It is very challenging and highly important to improve the performance of wireless mesh networks so that the throughput scalability of such networks can meet the needs of different users. One way to achieve significantly better performance for wireless mesh networks is to apply a technique called network coding [52]. Unlike in conventional networks, in wireless networks using network coding, intermediate nodes do not store and forward the same packets as sent

by the source node. In stead, intermediate nodes forward new coded packets computed by themselves from the packets they have received [56, 57, 58, 59, 60]. Hence, the data is actually mixed at each intermediate node before it is forwarded. This idea of mixing data at intermediate nodes takes advantage of the broadcast nature of wireless transmissions, and achieves great performance gains.

Many wireless mesh networks have user contributed wireless devices as their nodes. Since users normally have their own interests, economic incentives become a crucial problem. A selfish or economically rational user may let her wireless device deviate from the communication protocol, as long as the deviation is beneficial to herself. However, this deviation may harm the network's performance, or even lead the network to stop functioning in the worst case. Therefore, a lot of work has been done so far using payment based scheme to make the communication protocol incentive compatible, so that nodes have incentives to faithfully follow the protocol.

In a payment based system, a source node (or, sometimes, a destination node) is required to pay a certain amount of virtual money to intermediate nodes who transmit its data. Correspondingly, nodes that help transmitting other nodes' data can earn virtual money. Such a scheme will efficiently save the network resources and improve the efficiency of data transmission, since the source node can not transmit its data unlimitedly and intermediate nodes are pleased to provide their communication resources for forwarding other nodes' messages. In addition, under well designed payment schemes, a node may find that being selfish will hurt its own benefit and thus have incentives to be honest.

Nevertheless in reality, the node in charge of payment calculation may be

selfish or corrupted. It may miscalculate the payment for its own good or collude with others for the benefit of a cheating group. For example, if the source is responsible for calculating the payment for each intermediate nodes (which is quite common as the source collects routing information in many routing protocols), it may decrease the payment deliberately to save its own virtual money. Or in another case, if the payment is calculated by an intermediate node, the payment due to this intermediate node is calculated higher than it should be in order to benefit the intermediate node itself. This can lead to serious problems especially when there is no authority entity available for real time payment inspection. The entire system may collapse because of these selfish behaviors. Therefore we need a cheating detection scheme to fight against the misbehavior of payment calculator. As far as we know, no previous research has provided a general solution to this problem.

2.2 Cheating Detection for Payment Based Incentives with Application to Network Coding

2.2.1 Background and Motivation

In multi-hop wireless networks, users are inclined to serve for their own interests during data transmissions because there are limited communication resources. Economic incentives therefore become crucial problems in wireless networks. A selfish or economically rational user may let its wireless device devi-

ate from the communication protocol, as long as the deviation is beneficial to itself. On the other hand, this deviation may harm the network's performance, or even make the network stop functioning in the worst case. Therefore, we need to make the communication protocol incentive compatible, so that nodes of wireless networks have incentives to faithfully follow the protocol.

A major approach to fight against selfish behaviors in wireless networking is payment based scheme [72][73][74][75][76][49][25]. In a payment based system, a source node (or, sometimes, a destination node) is required to pay a certain amount of virtual money to intermediate nodes who transmit its data. Correspondingly, nodes that help transmitting other nodes' data can earn virtual money. Such a scheme will efficiently save the network resources and improve the efficiency of data transmission, since the source node can not transmit its data unlimitedly and intermediate nodes are pleased to provide their communication resources for forwarding other nodes' messages. In addition, under well designed payment schemes, a node may find that being selfish will hurt its own benefit and thus have incentives to be honest.

Nevertheless in reality, the node in charge of payment calculation may be selfish or corrupted. It may miscalculate the payment for its own good or collude with others for the benefit of a cheating group. For example, if the source is responsible for calculating the payment for each intermediate nodes (which is quite common as the source collects routing information in many routing protocols), it may decrease the payment deliberately to save its own virtual money. Or in another case, if the payment is calculated by an intermediate node, the payment due to this intermediate node is calculated higher than it should be in order to benefit the intermediate node itself. This can lead to serious prob-

lems especially when there is no authority entity available for real time payment inspection. The entire system may collapse because of these selfish behaviors. Therefore we need a cheating detection scheme to fight against the misbehavior of payment calculator.

To be concrete, we construct our system in a wireless network using MORE [17], a network coding system. The reason is that: First, network coding technique provides more efficiency in data transmission compared to traditional transmission protocols [17][11][12][13][58][59], and becomes more and more popular in current wireless networks. Second, we notice that although some existing protocols have been proposed to address incentive issues in multi-hop wireless network [25][26], which uses a trusted third party (TTP) to compute payment, they are not applicable to some scenarios , e.g., the network coding scenario. Specifically, with a network work coding system like MORE, it is impossible for a forwarding node to submit evidence for its forwarding, because such evidence does not exist [48]. When there is a dispute regarding forwarding, the only way to resolve the dispute is to survey other nodes in the neighborhood to see which is honest and which is cheating. In [48], a payment based incentive scheme is presented for such network, based on which we demonstrate our scheme in the following sections. One may suggest another approach based on TTP, in which the TTP initializes a threshold decision process once there is a dispute. We emphasize that this approach is not applicable as well, because in our settings the only available TTP is the central bank, and it may not be always online. Therefore, the objective of this work is to provide a general solution for payment cheating in multi-hop wireless networks without needing a payment inspector constantly online.

To solve the problem, we present a payee initiated threshold scheme to detect miscalculated payments based on threshold cryptography [3]. The scheme guarantees that, as long as the number of honest nodes is more than a threshold value, cheating behavior will definitely be detected and cheating nodes will be punished. Without loss of generality, in the rest of this work we always assume the source node is responsible for calculating payments unless otherwise specified. In our scheme, the source node publishes the calculation result of payments once the routing decision is made. When an intermediate node finds its payment miscalculated, it can start a threshold decision session by asking other nodes to recalculate its payment. When sufficient nodes (more than threshold) support its claim that the payment is miscalculated, the decision will be reported and the source will be punished even if the source may collude with some other nodes.

In this work, we made following contributions:

- We are the first to provide a general solution to payment cheating in wireless networks without needing a payment inspector constantly online.
- We present a cheating detection scheme in which a node can initiate a threshold decision session when it considers its payment miscalculated.
- Experiment results show that our scheme is efficient.

2.2.2 Secret Sharing Scheme

In reality, sometimes we want a secret to be securely held by different people, because the secret is easy to be lost or abused when held by only a few people without inspection. On the other hand, intuitively we can not share the secret

with too many people. In [1], a well known scheme called Shamir secret sharing is presented. In this scheme, a secret S may be divided into n different parts $s_1, s_2, ..., s_n$. Each part is called a secret shadow and is distributed to one party. Any single shadow (or any combination of fewer than t shadows) contains no information about the secret. Only if people own at least t shadows (t is a parameter determined before shadow generation), can they recover the secret. In their scheme, the shadow holders should keep their own secret shadows secure and share them in regulated procedures. This ensures that the secret is recovered only when at least t shadow holders come to the consensus to recover the secret.

The scheme works as follows: To divide the secret S into n pieces, we pick a $(t-1)$ degree polynomial $P(x) = c_0 + c_1 x + c_2 x^2 +, \ldots, +c_{t-1} x^{t-1}$ in which $c_0 = S$ and $(c_1, c_2, ...c_{t-1})$ are random coefficient numbers, and then get a secret shadow set $K = \{k_1, k_2, ..., k_n | k_i = P(i), 1 \leq i \leq n\}$. Given any subset of t of set K, we can recover the original polynomial, therefore obtain the secret $S = c_0 = P(0)$.

2.2.3 Cheating Detection Scheme

In order to explicitly show the mechanism of our scheme, we assume that our scheme is implemented on top of one payment scheme for MORE [48]. To make it complete, we briefly review the fundamental elements of their payment scheme.

In [48], Wu, et al., assume each data packet has a size of L and that to transmit a packet of size 1 takes one unit of cost. In MORE protocol, we can get the number of transmission for each packet z_i for node i from function

$z_i = f(S, D, i, \{(i, j, \epsilon_{i,j}) | (i, j) \in E\})$ Here the payment to node i is calculated as follows:

$$p_i = z_i L + \sum_{(i,j) \in E} \alpha(1 - \epsilon'_{i,j}),$$

where $\epsilon'_{i,j}$ is the received loss probability of link (i, j) from node i, L is the packet length, and α is a small parameter chosen by the administrator.

The calculated payments are reported to a central bank, where payment for each node is recorded after every successful data transmission session. In [48], it has been proved that it is a strict dominant strategy equilibrium for all nodes to truthfully report their routing information and follow the scheme.

However, the central bank can be offline and can not provide real time payment inspection for each transmission session. Nodes may not receive immediate help from central bank when they consider their payments miscalculated. In order to effectively detect the miscalculated payment, if node i finds its reported payment p'_i miscalculated such that $p'_i < p^*_i$, where p^*_i is its deserved payment, node i can start a threshold decision session to announce that its payment is miscalculated and collect supports from other nodes. Any other node should first recalculate node i's payment based on the routing information they share (e.g. loss probability matrix M_ϵ in MORE), and then reply node i with its own secret shadow if it agrees (we will introduce how this secret sharing scheme works in the other subsection). Once node i collects sufficient shadow information from others, it can decrypt the certification message with which its payment recalculation request will be accepted by the central bank.

In our scheme, in addition to manage payments, a central bank also acts as an authority entity to support the threshold scheme. The central bank will create secret shadow k_i for node i and publishes encrypted certification messages

for different threshold decision sessions. It also verifies the decrypted certification message submitted by the nodes which have collected sufficient support through a threshold decision session. Moreover, it will exclude the cheating node from the network once misbehavior is identified and reallocate payments if necessary.

We assume that the communications between the central bank and any other node use reliable links, such that the decision result will safely arrive at the central bank. By introducing a threshold scheme into wireless networks with network coding, we can resolve the possible disagreement in payment between nodes. Compared with voting mechanism, threshold decision in payment cheating detection can greatly decrease the workload of central bank. In a voting system in which a payment recalculation request is verified through checking digital signatures of all supporting nodes, the central bank can not determine whether the payment recalculation request is acceptable until most of the digital signatures have been checked, which is quite time consuming. In contrast, in our scheme, the recalculation decision is determined by nodes themselves before being reported to the central bank. Through a payee initiated threshold decision session, a node can decrypt the certification message as long as it collects sufficient support for its recalculation request. Upon receiving the threshold decision result, the central bank is confident that the session result has been confirmed and supported by the majority of nodes and it only needs to verify the certification message before the recalculation request is accepted. Therefore, we develop the threshold decision mechanism into our scheme.

We divide the whole procedure into two parts: certificate processing (Algorithm 1) and threshold decision (Algorithm 2). The first part is mainly per-

formed at central bank side, where the secret shadow for each node is generated and certification message is verified. To be more efficient than traditional secret sharing schemes, the central bank only needs to construct a polynomial once to obtain secret shadow for each node, moreover we use different primitive generators in order to keep the secret shadows independent from each other in different sessions. This can greatly improve the efficiency of the algorithm, because we do not need to construct different polynomials for different threshold decision sessions. In the second part, nodes are allowed to request their payments being recalculated to defend against selfish behaviors.

In shadow generation session, the central bank computes two types of important messages for all nodes in the network: shadow and certification message. The shadows should be distributed to all the nodes only *once*. Each node should secretly hold its shadow and repeatedly use it for different sessions. To compute the shadows, the central bank first randomly picks a polynomial P and uses $P(0)$ as the secret. Then the bank computes $K = \{k_1, k_2, ..., k_n | k_i = P(i), 1 \leq i \leq n\}$, and distributes each shadow information k_i to node i. After that, the central bank creates the encrypted certification message for each session, and publishes them to the network. The certification message is used for each threshold decision session, and cannot be repeatedly used. This is the end of its initialization step. We argue that the computation can be done in advance on central bank side whenever it's available even before the central bank goes online. Although the central bank does distribute shadows for each session, it is not necessarily involved in each session. Thus the overall overhead can be greatly reduced.

Every time the central bank communicates with all the network nodes, it will

collect the threshold decision session result including the certification message from node that receives sufficient support for payment recalculation request. The session result consists of two parameters and is easily to be verified: the first parameter indicates the sequence of data transmission session, and the second parameter is the certification message recovered from secret shadows. With this information, the bank understands that the node has already collected at least t (suppose t is the threshold) shadows. If the certification message is verified, the central bank will accept the payment recalculation request of corresponding node and punishes the dishonest source.

In the second part of our scheme, the threshold decision session is initiated by nodes themselves regardless of whether they can communicate with central bank in real time. Before data transmission, each node may receive its future payment for forwarding other nodes' packets from source. If a node finds its payment miscalculated, it can start a session to ask other nodes to recalculate its payment. To explicitly claim its request, this node should broadcast his new recalculated payment value with transmission session sequence and its node sequence. This is phase 1 of Algorithm 2. In phase 2, upon receiving the threshold decision request, a node should calculate the corresponding payment based on the shared loss probability matrix. If the result is consistent with the claimed one, the node should reply with its own modified shadow information and node sequence. As long as the session initiator has collected at least t shadow information, it can recover the original certification message and send his payment recalculation request to the central bank (see Algorithm 2).

2.2.4 Security Analysis

Threat and Trust Models

In wireless networks, we may have different kinds of attacks from participant nodes. Selfish nodes may cheat for their own benefits, while malicious nodes may deliberately disobey rules to deteriorate the networks. Although security tokens are useful in fighting against such misbehavior in traditional packet forwarding schemes, we notice that they are not applicable to some other network protocols, e.g. network coding. In this work, we focus on defending against the misbehavior of the node in payment dispute in network coding scenario, no matter whether this node is selfish or malicious. Therefore, in our threat models, a misbehaving node might be a selfish source which tries to decrease payments to others in order to save its own benefit, an irrational node which computes false payments to cause network disorders, or an intermediate node which falsely claims its payment is mistakenly computed.

Moreover, those misbehaving nodes may collude with each other rather than cheating alone. For example, a source and some intermediate nodes collude to cheat in order to decrease other nodes' payments and increase their overall benefits. Or some intermediate nodes collude to claim their payments are mistakenly computed by the source while they are not.

In our trust models, each node fully trusts the central bank to perform billing and authentication. The central bank trusts payment calculator if there in no payment dispute, but once there is any dispute, the central bank only trusts the node with the valid certification message of that threshold session.

Defenses

Suppose we have a set V of n nodes in a wireless network. Denote by N_c the set of those cheating nodes, where $N_c \subseteq V$. Also define $P_d^* = (p_d^*(1), p_d^*(2), ..., p_d^*(n))$ as the correct payment set for dth transmission session and $p_d^*(i)$ is the correct payment for node i.

We show that as long as the threshold $t > n/2$, our scheme will always detect misbehavior if no more than $(n - t)$ nodes are selfish or corrupted. If there are nodes cheating in our scheme, there are two possible cases:(1)the node in charge of payment calculation cheats; (2)the node in charge of payment calculation does not cheat.

In the first case, assume the payment set for dth transmission session will be P_d' such that $\exists p_d(i) \in P_d'$ and $p_d(i) \notin P_d^*$. If $\forall p_d(i), i \in N_c$ and the source $S \in N_c$, then the total benefit of the cheating group will be 0, as the source has to pay the same number of total amount of payment. Thus cheating nodes have no incentives to cheat in this way. If $\forall p_d(i), i \in N_c$ and the source $S \notin N_c$, then the source S is honest and can start a threshold decision session. If $\exists p_i(j) \notin N_c$, then node j is honest and can start a threshold decision session to claiming his deserved payment to be $p_i^*(j)$. Regardless of which node starts the threshold decision session, that node will collect at least t shadows and digital signatures from honest nodes, because no more than $(n - t)$ nodes cheat, which will beat the cheating nodes in numbers when reporting to the central bank about session result.

In the second case, the payment set will be P_d^* for dth transmission session. Although a cheating node i may claim his new payment $p_d'(i)$, such that $p_d'(i) > p_d^*(i)$ and receive support from other cheating nodes, we know from our basic assumption that node i will collect less than t shadows because $(n - t) < t$ and

will never recover the certification message by colluding.

Our scheme can effectively defend against attacks in which a node reuses the certification message or a node requires higher payment than it deserves. Because when cheating behavior is detected, the payment is recalculated neither by the selfish source nor by the recalculation requestor node, rather the payment will always be recalculated by a more reliable entity chosen by the administrator. Even if the recalculation result is not honest, as long as the honest nodes are more than half we can always get the payment correctly calculated by performing our threshold decision scheme repeatedly.

2.2.5 Evaluations

In this section, we perform three sets of experiments to evaluate our scheme using GlomoSim network simulator. First, we measure the overall overhead of Algorithm 1 and the overhead of Algorithm 2 in two phases in a simulated wireless network using network coding protocol MORE. We want to show that the efficiency of our scheme is reasonable and acceptable. Next, we measure and compare the overhead of our scheme with the overhead of Sprite [25] protocol in a simulated network to which the Sprite is applicable. Through the comparison, we want to show that although our scheme is slower than Sprite in some scenarios, the efficiency is still acceptable. Third, we measure and compare the computation overheads in Algorithm 1 when we set different threshold values. In this set of experiments, we want to demonstrate how threshold values affect the performance of our scheme. The general experiment parameters are set as follows:

- Experiment Settings

Grid	800m x 800m
Radio transmission range	150m
Simulation duration	100s
Node placement	Uniform
Mobility	Random waypoint
Bandwidth	2Mbps
Packet length	32Byte
Traffic type	CBR
Background throughput	500Kbps

Note that in GlomoSim when random waypoint model is used, a node randomly picks a destination and moves towards the destination in a speed uniformly chosen in a range [Min_Speed, Max_Speed]. After it reaches its destination, the node waits there for a period of time and then moves to another destination. In our simulation, we set Min_Speed = 0, Max_Speed = 10 m/s, and the waiting time to be 30s.

Overall Overhead in Networks using Network Coding

In the first set of experiment, we simulate up to 50 nodes in a wireless network using network coding protocol MORE, and let node 1 be the central bank which computes the shadows and certification messages for all other nodes. We randomly pick a pair of source and destination and let the source send CBR flows to the destination. For the purpose of simplicity, we guarantee the connectivity between each node in our simulation. The payment for each intermediate node is computed by the source following [51] and the amount of payment is

inspected by the intermediate node itself. In addition, we assume the threshold value is 0.8 in this set of experiment.

We first evaluate the overall overhead of Algorithm 1 that includes the computation overhead and the communication overhead. The computation overhead in Algorithm 1 consists of two parts: the overhead used to compute the shadows and the certification messages and the overhead used to validate the collected information and punish the cheating node. Here in this evaluation, we assume the central bank generates certification messages for one threshold session. On the other hand, the communication overhead depends on how many participant nodes there are in the network. The reason is that the central bank has to communicate with each node individually. Note that the overhead of payment computation and link state report should not be considered in this measurement. The results are shown in Figure 2.1. We can see the overall overhead of Algorithm 1 grows linearly as the number of nodes increases. If there are no more than 50 nodes, the central bank takes less than 20 ms to perform Algorithm 1 for one threshold session.

Then we evaluate the overhead of two phases in Algorithm 2. In this evaluation, we also use the first experiment settings as above. Phases are determined based on different roles of nodes in a threshold session: In phase 1, we simulate the scenario in which a node starts a threshold decision session and broadcasts its request to other nodes. In phase 2, each node receives the request generated in phase 1 and replies if it agrees with the initiator. Also the session initiator should try to retrieve the certification message in phase 2. In this evaluation, the computation overhead mainly depends on the efficiency of retrieving the certification message. And the communications in Algorithm 2 are round trips

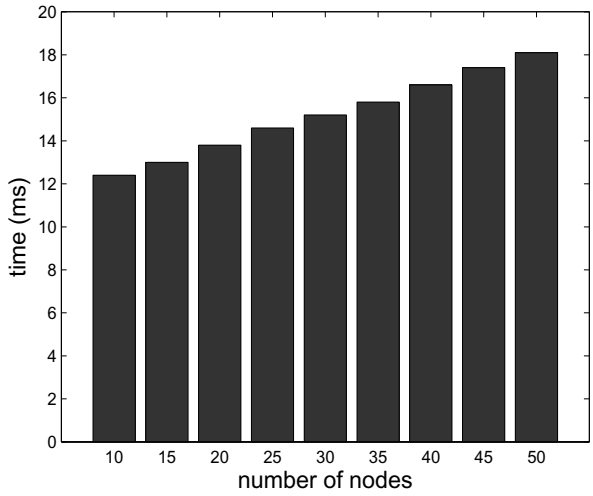

Figure 2.1. The overall overhead of Algorithm 1 in Section 2.2.

between one specific node (session initiator) and all other nodes. The results are shown in Figure 2.2. Similarly like the overhead of Algorithm 1, the total overhead of two phases in Algorithm 2 grows linearly as the number of nodes in the wireless network increases. When less than 50 nodes are deployed in our experiment, the total overhead of Algorithm 2 is less than 50 ms. In phase 1, the overhead is no more than 8ms for up to 50 nodes. In phase 2, the overhead is no more than 40ms for up to 50 nodes. Furthermore, we can observe that the percentage of the overhead in phase 1 of Algorithm 2 is below 20%.

Our Scheme v.s. Sprite

In this set of experiment, we compare the efficiency of our scheme with that of an existing protocol namely Sprite [25]. As we have mentioned, Sprite is not applicable to all network scenarios, e.g. opportunistic routing. Therefore, we use traditional routing and packet forwarding schemes in our second set of experiment to fit both schemes. We assume there are 50 nodes in the simulated network, and let the sender send up to 20 messages to the destination. The

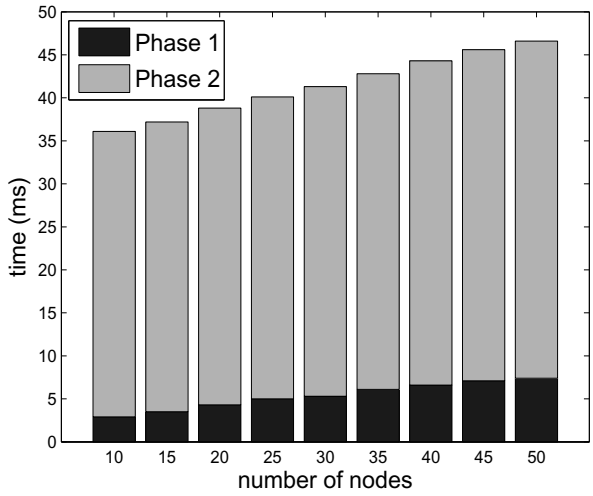

Figure 2.2. The overhead of phase 1 and 2 in Algorithm 2 in Section 2.2

payment for each intermediate node is determined based on the length of the forwarded packet in our scheme, and we let the sender mistakenly compute a payment of one message for one intermediate node during the simulation.

To evaluate Sprite protocol, we follow the experiment settings in [25] except that the network topology is the same as in our first set of experiment. In order to estimate the overhead of the cryptographic computation, we use Crypto++ v5.6 library for calculation in a laptop with an Intel processor at 2.4 GHZ and 2 GB RAM.

The results are shown in Figure 2.3. The overhead of our scheme does not increase if the number of the sent messages increases. However, the overhead of Sprite linearly increases if the number of the sent messages increase. The reason is that both algorithms are not designed to secure each message, but for the entire session. Instead in Sprite, the overhead is introduced to the system for each transmitted message. Each message and its corresponding receipts have to be secured through cryptography. Therefore, we can conclude that in scenarios

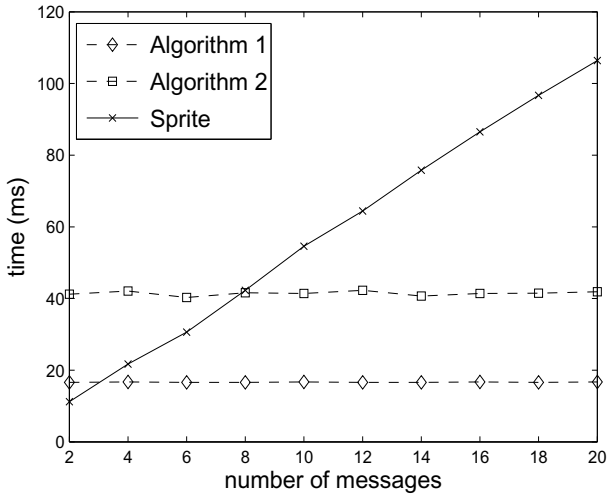

Figure 2.3. Overhead comparison between our scheme and the Sprite protocol in Section 2.2.

where a large number of messages are to be transmitted and the sender cheats occasionally, the efficiency of our scheme will be better. We also notice that the overheads of our algorithms are slight different from that in the first set of experiment, it is because the network topology has changed.

Computation Overhead

In this subsection, we want to show how threshold value affects the computation overhead of our scheme.

Computation overhead includes two parts in our scheme: shadow processing and certification message encryption/decryption. In Algorithm 1, the central bank computes secret shadows for each session and the overhead of shadow generation depends on the threshold value and the total number of nodes. While in Algorithm 2, before the certification message is decrypted we only need to compute a_i, which takes much less time than shadow generation overhead in Algorithm 1. In both algorithms, the certification message encryption/decryption

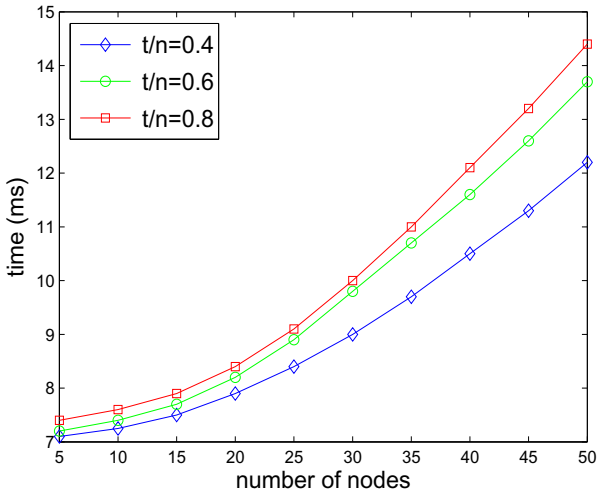

Figure 2.4. Computation overhead in Algorithm 1 in Section 2.2.

overhead is relatively constant as the length of the certification message does not change. In the following experiment, we measure the computation overhead of Algorithm 1 and 2 if there is only one certification message published during each session and the result can illustrate the relationship among computation overhead, the threshold value and the total number of nodes.

We first measure the computation overhead in Algorithm 1, and the results are shown in Figure 2.4, where t/n is the ratio of the threshold value to the total number of nodes. The greater the ratio is, the greater the computation overhead of Algorithm 1 we get in the evaluation. Also the computation overhead grows as the total number of nodes increases, however it is still less than 15 ms.

Computation Overhead in Algorithm 2 : 30.23 ms. This amount of time is needed for a node to recover a certificate using sufficient secret shadows. It is worth mentioning that the computation overhead of Algorithm 2 is constant regardless of the threshold value and the total number of nodes, because to compute a_i requires much less time than to create secret shadows from a polynomial

with random coefficients. The certification message decryption procedure takes up most of the computation overhead in Algorithm 2.

2.2.6 Summary

In this work, we present a threshold scheme to fight against the miscalculation of payment in payment based incentive wireless networks. We integrate the threshold cryptography into existing payment based schemes and demonstrate that nodes no longer have incentives to cheat in payment calculation. The experiment results show that our scheme is efficient. Since our scheme is designed to work in a reliable channel, one open problem is how to implement threshold scheme in an unreliable environment like in lossy links.

2.3 An Incentive Scheme for Packet Forwarding and Payment Reduction in Wireless Networks using XOR Network Coding

2.3.1 Background and Motivation

XOR network coding [59] exploits the shared nature of the wireless medium which broadcasts packets in its neighborhood during transmission. Each node stores the overheard packets for a short time. It also shares the information of which packets it has heard with all its neighbors. Before transmitting a packet,

a node uses its knowledge of what its neighbors have heard to perform opportunistic coding. The node XORs as many native packets as possible into a single encoded packet, and transmit it if each intended next hop has enough information to decode the XOR-ed packet. Through maximizing the number of native packets delivered in a single transmission by each node, the overall throughput is improved.

In practice, there are always some costs (e.g. energy consumption, transmission overhead, etc) induced to a node when it helps forwarding packets for others. If nodes have their own interests (especially in user contributed networks), economic incentives become a crucial problem. Because a selfish intermediate node may deviate from the protocol as long as the deviation is beneficial to itself. In addition, this incentive problem may lead to the failure of the XOR network coding protocol, and the deterioration of the system throughput. Therefore, we need to make the XOR protocol incentive compatible, so that each node has incentive to truthfully follow the XOR protocol.

We note that the opportunistic routing scheme is used in XOR protocols like COPE, and the major objective of this work is to solve the incentive compatible *packet forwarding* problem in such networks. Although there are some incentive packet forwarding schemes that have been proposed for traditional wireless networks and randomized linear network coding networks [20]-[25], we emphasize that these existing schemes can not be used in XOR protocols, because they study the intra-session incentive compatibilities while XOR network coding is designed to exploit coding opportunity in context of inter-session. In intra-session network coding, coding is restricted to packets belonging to the same session, and in inter-session coding, coding is allowed among packets be-

longing to possibly different sessions. These existing schemes aim to provide incentives for intra-session network coding protocol e.g. MORE [17], however did not consider the scenarios of inter-session network coding.

In this work, we first present a simple but powerful payment scheme that guarantees each node will follow the XOR protocol. In this scheme, each source node has to pay an amount of credits to the intermediate nodes to cover the forwarding cost, such that the intermediate node is willing to forward the packets for the source. The difference between our scheme and other existing schemes is that our scheme considers the nature of XOR network coding in packet forwarding and provides incentives to each node for XOR-ing packets whenever possible. We can rigorously prove that under this payment-based scheme, it is a dominant strategy for each forwarding node to follow the XOR protocol.

Moreover, we consider the overpayment of source node in our basic payment scheme, and propose an enhanced payment scheme using Support Vector Machine (SVM) [27] model to reduce source payment. We know in XOR protocol, intermediate nodes are required to XOR packets in different sessions. For an intermediate node, the real cost of forwarding packets can be reduced if multiple packets are XOR-ed into one before transmission. The reason is that the node only needs to forward one encoded packet instead of multiple native packets.

However, in distributed networks the source node can hardly know how many times the packets has been or will be XOR-ed along their forwarding path. Although the source node may require each intermediate node to report its real cost of forwarding, these intermediate nodes can still cheat in reporting and demand false payments. Without knowledge of real cost of each intermediate node, the source node has always to assume that it has experienced the worst

case (no packet has been XOR-ed throughout the session) when paying credits for forwarding. Otherwise the amount of payment is insufficient once the worst case happens, which leads to the failure of the payment scheme. In other words, the basic payment scheme can not benefit from the nature of XOR network coding.

Meanwhile, for many nodes that need to make payments, such overpayments can be undesirable or even unaffordable. Therefore, we need an enhanced scheme based on our payment-based scheme to lower the source payment to a reasonable level in wireless networks using XOR network coding.

To address this problem, we propose an enhanced payment scheme using SVM model, in which source node exploits historical data through a training process and consequently achieves a payment prediction. The amount of payment can be reduced to a reasonable level.

Our contributions can be summarized as follows:

- We design a basic payment scheme under which each intermediate node has incentives to follow the XOR protocol. And we rigorously prove that to maximize the utility, the intermediate node has to XOR as many as packets as possible in our model.

- We also consider the overpayment problem and present an enhanced payment scheme, in which source node can largely reduce its payment using SVM learning.

- We perform extensive simulations and the results demonstrate that, under our schemes, the intermediate node will always harm itself if it deviates from the XOR protocol. Moreover, the results show that our enhanced

scheme can reduce the source node's payment to a reasonable level and make it close to the real forwarding cost.

2.3.2 Basic Payment Scheme

Suppose there is a source node S which chooses a set of n intermediate nodes $I = \{1, \ldots, n\}$ along its transmission path. We assume that forwarding packets will induce costs to an intermediate node because of energy consumption, transmission overhead, etc. And we assume that the cost depends on the length of data: transmitting data of length 1 has β units of cost. Denote by L the total length of data need to be forwarded in a session. Thus the cost of intermediate node i to forward all data is $c_i(L) = \beta L$. The utility of node i in the entire session is

$$u_i(L) = p_i^s(L) - c_i(L) = p_i^s(L) - \beta L, \tag{2.1}$$

where $p_i^s(L)$ denotes the payment from S to i for forwarding data of length L.

We observe that each intermediate node i has incentives to forward data as long as

$$u_i \geq 0. \tag{2.2}$$

Given Equation (2.1) and (2.2), we know an intermediate node i will forward data of length L for S as long as the payment satisfies:

$$p_i^s(L) \geq c_i(L) = \beta L. \tag{2.3}$$

Therefore, the lower bound of payment $p_i^s(L)$ is βL. And the lower bound of overall payment (we call maximum payment in the rest of this work) for the entire session is

$$p_{max}^s(L) = \sum_{i=1}^{n} p_i^s(L) \geq \sum_{i=1}^{n} \beta L = \beta n L. \tag{2.4}$$

In practice, credits should be paid after data is successfully received by the receiver. Each intermediate node i should report a feedback to S, if i can decode all data from its upper hop $(i-1)$. Upon receiving this feedback report from i, S pays corresponding credits to $(i-1)$. To defend against cheating in reporting, we can use digital signature. Note that in order to reduce overheads of our algorithm, feedback reports can be bundled into one and sent back to S periodically, instead of immediately after the session finishes. Consequently the credits are paid periodically rather than in real time.

Theorem 1. *In our basic scheme, an intermediate node has incentives to follow the protocol to XOR packets whenever possible.*

Proof. Suppose an intermediate node i is involved in q sessions during time interval Δt. Without loss of generality, we assume these q sessions begin and finish within Δt. Suppose the strategy to follow XOR network coding protocol is s^*, and the utility of i when choosing strategy s is $u_i(s)$.

If i follows XOR network coding, we know i will XOR as many packets as possible. Hence whenever i deviates from the protocol, it XORs less data than it should. In other words, some extra cost is induced since these data should be forwarded in a new transmission. Suppose i chooses a strategy $s' \neq s^*$ and thus deviates from the protocol $d(s')$ times during Δt. We denote c_0 as the forwarding

cost of i for all q sessions following s^*. We denote $c_i^j(s')$ as the extra cost induced to i in its j-th deviation under strategy s'. Simply we know

$$c_i^j(s) > 0, \forall 0 < j \leq d(s). \tag{2.5}$$

Moreover, in our basic scheme the payment for i in each session is fixed. During Δt, the utility of i when choosing strategy s' is

$$\begin{aligned} u_i(s') &= \sum_{j=1}^{q} p_i^j - (c_0 + \sum_{j=1}^{d(s')} c_i^j(s')) && (2.6)\\ &= \sum_{j=1}^{q} p_i^j - c_0 - \sum_{j=1}^{d(s')} c_i^j(s') && (2.7)\\ &= u_i(s^*) - \sum_{j=1}^{d(s')} c_i^j(s'). && (2.8) \end{aligned}$$

Therefore, given Equation (2.5) we have

$$u_i(s') < u_i(s^*), \forall s' \neq s^*. \tag{2.9}$$

□

2.3.3 Enhanced Payment Scheme using SVM model

Although the basic payment scheme guarantees that all intermediate nodes follow XOR protocol truthfully, we still need to address the overpayment problem in order to make our scheme efficient and practical.

In distributed wireless networks, a source node can hardly have complete knowledge about the entire network, such as network topology, active sessions,

etc. Information exchanging and sharing among nodes are always limited and inadequate. Hence it is impossible for a source node to trace all the packets along the forwarding path, or to calculate the real forwarding cost of each intermediate node precisely. In addition, although source node may require intermediate nodes to report their real cost of forwarding after session transmission, those intermediate nodes may not truthfully report for the sake of better payments, therefore can not be trusted by source node.

To address this problem, we propose an enhanced payment scheme using SVM model [27], in which source node exploits historical data through a training process and consequently achieves a payment prediction. The amount of payment can be thus reduced to a reasonable level.

One major advantage of SVMs is that it can deliver a unique and global solution, which can not be achieved by other training models such as Neural Networks. Moreover SVMs avoid overfitting by choosing the maximum margin separating hyperplane in a higher dimensional space than the input space. The complexity of calculations in SVMs does not depend on the dimension of the input space but on the number of support vectors, which makes it more efficient than others when we have inputs with large dimensions.

In our system, the forwarding cost of node i is closely related to i's frequency of XOR-ing packets. Suppose an intermediate node serves for two sessions, and the two sessions have the same packet length and transmit at the same rate. If this node XOR all packets of these two sessions, then the forwarding cost of this node is halved through XOR coding. The more an intermediate node does XOR coding, the less forwarding cost it takes. The chance of XOR coding depends on the number of sessions this intermediate node involves concurrently. Therefore,

we focus on the information of concurrent sessions each intermediate node i involves.

We model the input data vector for SVM as $x = (f_1, f_2, \ldots, f_n)$, where n is the total number of nodes in the network and each component denotes the weight of a node. The weight indicates the activeness of the node. The higher the weight is evaluated, the more likely the node can make XOR coding. We define a function $f = \Omega(x; R_1, \ldots, R_x)$ to compute the weight of a node if this node is involved in x concurrent sessions. R_i is the transmission rate of session i. $\Omega()$ should be a strictly increasing function and determined by the administrator. Specifically, in our SVM model $(f_1 \geq f_2 \geq \ldots \geq f_n)$. In other words, $f_j (1 \leq j \leq n)$ denotes the j-th largest weight among all nodes. Therefore, the dimension of input data vectors is n.

Label y denotes the class that a training input x belongs to, and the input pair (x, y) is used to train the SVM model. The training data can be viewed as labeled points in an n-dimension input space V. In SVM model, the learning task is to find directed hyperplanes in V such that those points with the same label can be classified in the same separated space of V. The directed hyperplane found by a SVM is intuitive: it is the hyperplane which is maximally distant from the classes of labeled points located on each side. Thus, the closest points on both sides have most influence on the position of the separating hyperplane, and are therefore called support vectors. The distance between the support vector to the separating hyperplane is called the *margin*. SVM technique is to find a separating hyperplane which can provide the largest margin in order to minimize the generalization error. The separating hyperplane is given as $w \cdot x + b = 0$, where $\cdot$ denotes the inner product, w determines the orientation and b is the offset of

the hyperplane from the origin in V.

In an example of two-class SVM model with data $x_i (i = 1, \ldots, m)$ and $y_i = \pm 1$, and let the decision function be

$$f(x) = sgn(w \cdot x + b). \tag{2.10}$$

We also implicitly define a scale for (w, b) by setting $w \cdot x + b = 1$ for the closest support vector to the separating hyperplane on one side, and $w \cdot x + b = -1$ for the closet support vector on the other side. Then the problem of maximizing the margin is equivalent to minimizing

$$\frac{1}{2}||w||_2^2, \tag{2.11}$$

subject to the constraints:

$$y_i(w \cdot x_i + b) \geq 1, \forall i. \tag{2.12}$$

As a constrained optimization problem, the above formulation can be reduced to *minimization* of the following Lagrange function:

$$L(w, b) = \frac{1}{2}(w \cdot w) - \sum_{i=1}^{m} \alpha_i (y_i(w \cdot x_i + b) - 1), \tag{2.13}$$

where α_i are Lagrange multipliers and $\alpha_i \geq 0$. Therefore we can compute w and b by taking the derivatives with respect to w and b, and further have our decision function $f(x)$ determined.

Given Equation 2.13, in order to compute the minimum value, we take the

derivatives of $L(w, b)$ with respect to w and b and set them to zero:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0 \tag{2.14}$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{m} \alpha_i y_i = 0. \tag{2.15}$$

Substituting w from Equation (8) back into $L(w, b)$, we get a new formulation:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j x_i x_j, \tag{2.16}$$

which should be maximized with respect to the α_i subject to the constraints:

$$\alpha_i \geq 0, \tag{2.17}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0. \tag{2.18}$$

In SVM, we know that the generalization error bound does not depend on the dimensionality of the space. Therefore if we substitute $x_i x_j$ with $K(x_i, x_j) = \Phi(x_i)\Phi(x_j)$ in Equation 2.16:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j), \tag{2.19}$$

which makes us easier to find a solution for the problem. In other words we map the data points in V into a new space V' (in most cases a higher dimensional space that we call feature space). We call this $K(x_i, x_j)$ a kernel function, while in our system we use the linear kernel $K(x_i, x_j) = x_i x_j$ as our kernel func-

tion (there are some other non-linear kernels, e.g. a Gaussian classifier when $K(x_i, x_j) = e^{-(x_i - x_j)^2/2\sigma^2}$, however linear kernel serves very well in our system and moreover faster than other kernels). By solving the problem in Equation (2.16) subject to (2.17)(2.18), we can obtain α_i and furthermore b. Thus for a test input x_s, decision z_s is the determined class for the test input x_s after training.

$$z_s = sgn(\sum_{i=1}^{m} \alpha_i y_i x_i x_s + b). \tag{2.20}$$

Suppose there are k classes in the model. Each class represents the percentage of maximum payment p^s_{max} that source node S can reduce to. Denote by $P(z)$ the percentage represented by class z. To build a k-class SVM model based on the knowledge of two-class model, we can do as follows: [19]:

1. for each incoming test data, we hold a decision list which is initialized with a list of all classes;

2. the test data is evaluated against the decision function that corresponds to the first and the last element in the list;

3. if the test data is determined to be in one of above two classes, the other class is eliminated from the list;

4. repeat step 2 and 3 for $(k-1)$ times until there is only one class in the list, then the test data belongs to this class.

Hence, a source node S may calculate its payment prediction using SVM model:

$$p^s_{svm} = P(z_s) p^s_{max}. \tag{2.21}$$

Before a source node begins transmission, it requires other source nodes to share their historical data and session information. In return, the source node will send its own data to those who are willing to share. If there exists a source node that does not respond to any data sharing request, then it may not get others' historical data either. Without help from others, this isolated source node can not reduce its payment and thus suffer from the overpayment problem.

The historical data about session path and payments are paired as training data for the SVM model. For example, if node i reports one of its historical session information x_i' and payment y_i' to source node S, then (x_i', y_i') is used by S as a training input in our SVM model. On the other hand, S should build its own test data based on the path information of all concurrent sessions. Source node first counts the number of sessions each node involves, then computes the weights of all nodes and obtains the test input x_s by sorting their weights. If S inputs x_s to the SVM model after training, it can have the decision z_s as an output.

After S gets the decision z_s, it has to further compute the payment for each intermediate node. Because the probability of each intermediate node to XOR packets varies, their forwarding costs are different. The source node should pay less to the intermediate node which has higher chance to make XOR coding. One possible solution is to divide the payment according to each node's weight: Suppose $g(i) = \frac{1}{f_i}$. Denote by $C(i)$ the number of concurrent sessions node i involves. The payment from source node S to intermediate node i is

$$p_i^s = \frac{g(i)}{\sum_{j \in I} g(j)} p_{svm}^s \tag{2.22}$$

$$= \frac{g(i)}{\sum_{j \in I} g(j)} P(z_s) p^s_{max}, \tag{2.23}$$

where I is the set of all intermediate nodes. Similarly, the payment for each intermediate node in the enhanced payment scheme may be cleared periodically.

The description of our enhanced scheme using SVM learning is shown in Algorithm 4

2.3.4 Evaluations

We integrate our schemes into XOR network coding protocol using Glomosim. Our experiments have two major objectives. First we want to illustrate that, under our basic scheme, an intermediate node has incentives to follow the XOR protocol. The other objective is to show that our enhanced scheme can effectively reduce source payments while ensuring the error prediction rate is acceptable. To achieve these objectives, we design three sets of experiments:

- In our first set of experiment, we focus on the utility of an intermediate node when this node behaves in five different modes when XOR-ing packets under our basic payment scheme. The node may choose to XOR packets in probability of $0\%, 20\%, 50\%, 80\%$ and 100%. The results show that the more packets it XORs, the more utility it gains. Also, we evaluate how these behaviors affect the system throughput.

- In the second set, we evaluate the effectiveness of the enhanced payment scheme. By comparing the maximum payment and the predicted payment of a random source, then comparing the predicted payment and the real cost of the same source, the results show that our enhanced scheme can

reduce the source payment to a reasonable level with an acceptable error rate (less than 5%).

- In the third set, we focus on the payment correctness for each intermediate node. The results show that our enhanced scheme let source node fairly distribute payments to each intermediate node. When we moderately increase the amount of payment prediction, the results show that the case of insufficient payment to intermediate nodes never happens.

In Glomosim, we consider a wireless network with 10 nodes using XOR network coding. In each test, we assume there are two types of data transmission sessions: background sessions and test sessions, where background sessions are active all the time and all test sessions begin and end simultaneously. The reason why we use background session is that we allow the background session to fully exploit the free bandwidth of links when we evaluate the system throughput. Thus we can measure the improved system throughput under the XOR protocol.

There are three kinds of transmission rate that are used in our evaluation: 1Mbps, 2Mbps and 4Mbps. All sessions use packets of length 1024 during transmission. And we define the weight function as $\Omega(x; R_1, \ldots, R_x) = \sum_{i=1}^{x}(\gamma(R_i))$, where $\gamma(1) = 1$, $\gamma(2) = 1.5$, $\gamma(4) = 2$. Also we use 10-class SVM model in the evaluation, while $P(i) = \frac{i}{10}$

Basic Scheme

We evaluate how intermediate nodes' different behaviors affect their utilities under our basic payment scheme. In this set of experiments, we generate 1 background session with transmission rate 2Mbps and 2 test sessions with

transmission rate 1Mbps and 4Mbps respectively. Each pair of source and destination is randomly picked.

First, we randomly pick 1 intermediate nodes from a test session in a transmission scenario, and observe its utility within this session. We repeat the transmission scenario for 5 times, while only the selected node changes its XOR coding strategies in each run, and other nodes always follow the XOR protocol. The selected node has five XOR coding strategies, and under each strategy the node XORs packets with a different probability. In our tests, the probabilities are 0%, 20%, 50%, 80%, and 100%. In other words, the selected node may never XOR packets although there are coding opportunities(in case of 0%), or may XOR packets whenever possible (in case of 100%). After that, we randomly pick 4 more nodes and do the same tests. The result is shown in Figure 2.5: Note that when the nodes do not XOR any packet (in case of 0%), the utilities are all 0. The reason is that the payment for each node is computed based on the cost when no packet is XOR-ed as in Equation (2.3). And if their probabilities of XORing packets increase, they gain more utilities. In these experiments, following the XOR network coding protocol will bring each intermediate node with the maximum utility under our basic scheme.

Then we repeat the above simulation again, and this time we evaluate their utilities of all involved sessions. Likewise, they have the same XOR strategies as in above tests. And we totally pick 5 random nodes to make these experiments. In these tests, we want to observe how the XOR strategy affects the overall utility of a node. In Figure 2.6, the result shows that they also gain 0 utilities when they do not follow the protocol at all. And still they gain more utilities if they choose to XOR more packets.

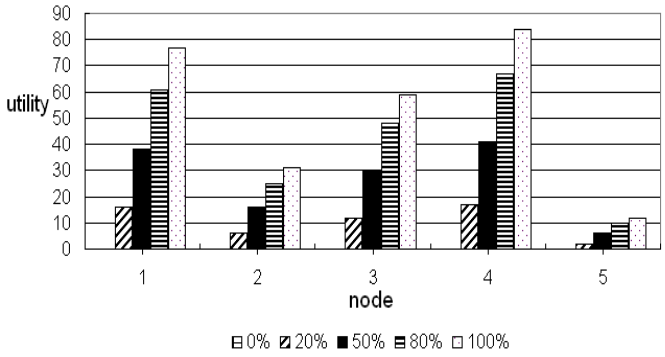

Figure 2.5. Comparison of utility changes in one session among 5 randomly picked intermediate nodes in Section 2.3.

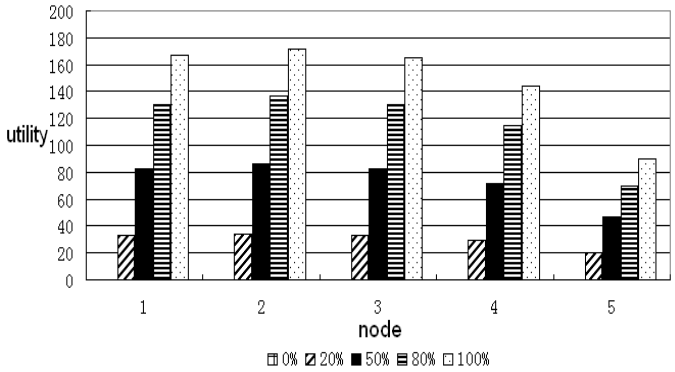

Figure 2.6. Comparison of utility changes in all involved sessions among 5 randomly picked intermediate nodes in Section 2.3.

In above experiments, we have shown that under our basic scheme, different XOR strategies lead to different nodes' utilities. Here we want to show that different XOR strategies will also affect the system throughput. Suppose all the nodes choose one of the three XOR strategies at the same time in this evaluation: 0%, 50% and 100%. To better illustrate, we allow the background session to increase its transmission rate whenever possible. The evaluation results of system throughput in shown in Figure 2.7, in which 100 results of each XOR strategy are represented in terms of cumulative fraction. We can see that when all the nodes follow the XOR protocol, the average system throughput is larger than

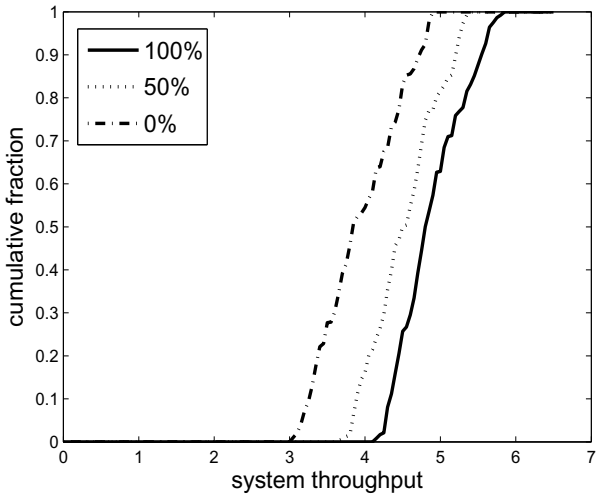

Figure 2.7. The cumulative fractions of system throughput when nodes follow three different XOR coding strategies in Section 2.3.

that when all nodes deviate.

Enhanced Scheme

In the second set of experiments, we generate random number of test sessions at random transmission rate (1Mbps, 2Mbps or 4Mbps). We randomly choose pairs of source and destination in each simulation. And Overall we run this simulation 100 times.

First, we assume that a randomly picked source node may choose to follow our enhanced scheme (in other words, willing to share its historical data and routing path information with other nodes) or not. Note that all other nodes will follow the enhanced scheme. We focus on the payment comparison between these two situations. In the first case, the source node can not use SVM model to predict a reduced payment, therefore has to choose a maximum payment p^s_{max}. While in the latter case, the source node can obtain a payment prediction p^s_{svm} after SVM training. We record the ratio $P_1 = \frac{p^s_{svm}}{p^s_{max}}$ in each run, where P_1

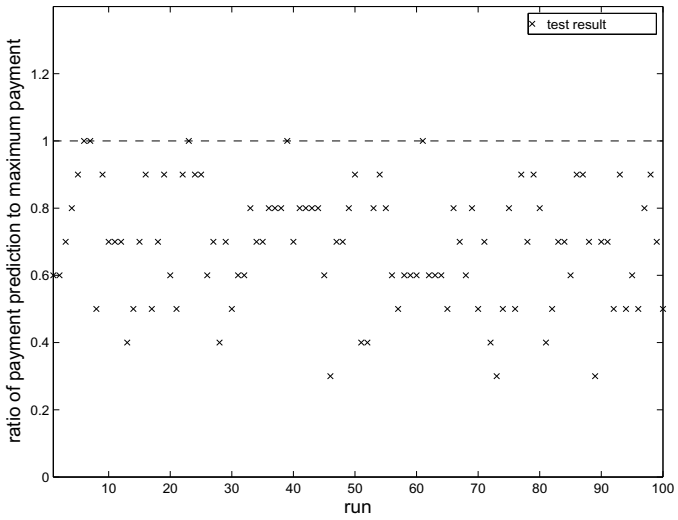

Figure 2.8. The ratio of the payment prediction to the maximum payment of a random source node in Section 2.3.

is actually the predicted percentage $P(z_s)$ of SVM. In Figure 2.8, we notice that $30\% \leq P_1 \leq 100\%$ in this set of experiments, which means the enhanced scheme can reduce the source payment by up to 70%.

In each run, we also compare the payment prediction p^s_{svm} and the entire forwarding cost for the test session c_s. we record the ratio $P_2 = \frac{c_s}{p^s_{svm}}$ and the results are shown in Figure 2.9. We can see that in over 80 out of 100 runs, P_2 lies in range of $(0.8, 1)$, which means that the payment predictions are close to the real costs and stay larger in order to provide positive utilities in these runs. In 2 runs, the payment prediction is insufficient to cover the cost $p^s_{svm} < c_s$. In this evaluation, the SVM model prediction error rate 2% is acceptable.

In Figure 2.8 and 2.9, we have shown that our enhanced scheme provide a reasonable overall payment prediction to the source node. In Figure 2.10 and 2.11, we want to show that each intermediate node will receive sufficient payment under the enhanced scheme.

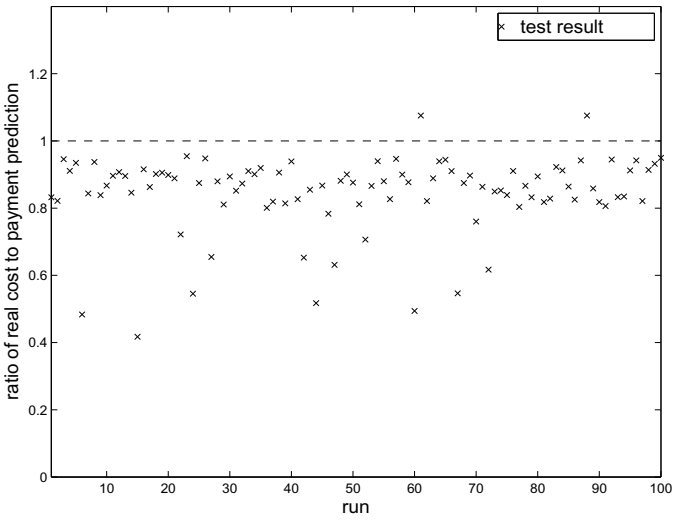

Figure 2.9. The ratio of the real forwarding cost to the payment prediction of a random session in Section 2.3

We design the third set of experiments as follows: In each run, we pick random number of pairs of source and destination. The sessions are test sessions and all transmission rates are randomly determined. Meanwhile, the enhanced scheme is implemented in the network.

We compare the payment to a random intermediate node with its real cost of forwarding. In Figure 2.10, we select 7 intermediate nodes and display the value of their payments together with their real forwarding cost. We can see that the payment is either slightly larger than or equal to the cost, which indicates that our enhanced scheme is good. In Figure 2.11, we study the ratio of real forwarding cost to its received payment of a random node in 100 runs. In this test, to ensure that all intermediate nodes receive sufficient payment, we moderately increase the class level of prediction result by 1. We notice that in most cases (97 out of 100 runs), the cost is slightly less than the payment, while in other 3 runs, the node takes much less cost in forwarding. Overall, the payment is always sufficient for a randomly picked intermediate node.

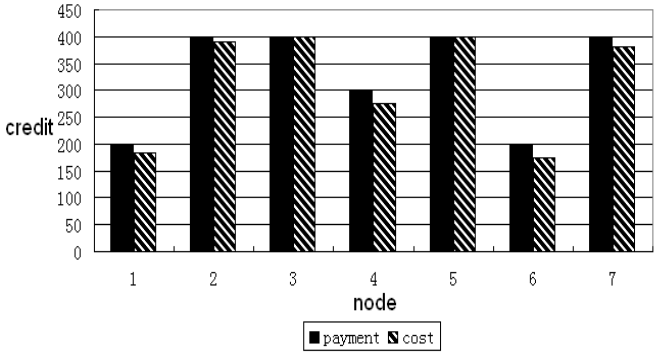

Figure 2.10. Comparison of the received payments among 7 random nodes in Section 2.3.

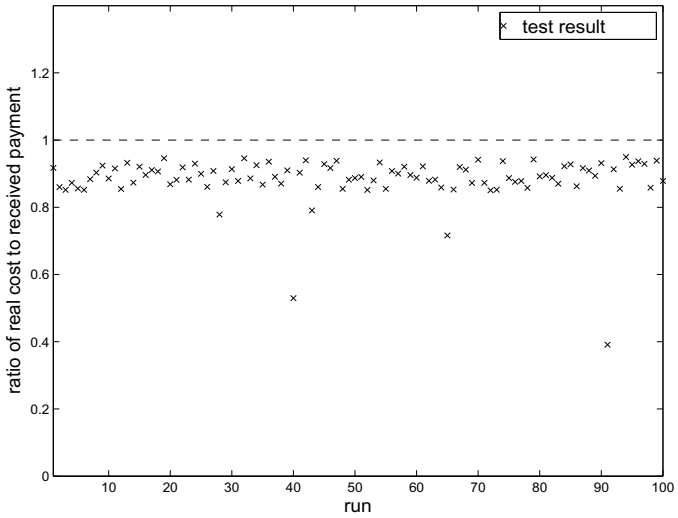

Figure 2.11. The ratio of the real forwarding cost to the received payments of a random node in Section 2.3.

Overhead

In the basic scheme, the computation complexity is $\Theta(1)$ and the communication between source and intermediate nodes is about feedback reporting and not required to be done in real time. Therefore, the overhead analysis in the basic scheme is trivial.

In our enhanced scheme, we build our SVM model using Matlab SVM tools. The communication overhead is induced when source and other nodes share

Table 2.1. Communication overhead of the enhanced scheme in Section 2.3

Overhead	N=10	N=20	N=30
Communication	63ms	77ms	94ms

Table 2.2. Computation overhead of the enhanced scheme in Section 2.3

N=10,D=100,k=5	N=10,D=500,k=5	N=10,D=1000,k=5
45ms	376ms	1.65s
N =10,D=100,k=10	N=10,D=500,k=10	N=10,D=1000,k=10
99ms	846ms	3.71s
N=20,D=100,k=5	N=20,D=500,k=5	N=20,D=1000,k=5
125ms	377ms	1.65s
N =20,D=100,k=10	N=20,D=500,k=10	N=20,D=1000,k=10
281ms	848ms	3.71s

historical data and session information, which can be done in a round-trip manner. Therefore this overhead is determined by the largest round trip delay between source and other nodes.

The computation overhead of the enhanced scheme is determined by the time consumed in SVM data training and predicting. In Table 2.1 and 2.2, we list the communication and computation overhead in different circumstances, where N denotes the number of nodes in the network, D denotes the size of the set of training data, and k denotes the number of classes in SVM model.

From Table 2.1, we can see that the communication overhead in the enhanced scheme is less than 100ms. The results in Table 2.2 show that the computation overhead in the enhanced scheme linearly increases by the number of classes k, and is mostly affected by the size D of the training data set.

2.3.5 Summary

In this work, we consider the incentive compatibility in wireless networks using XOR network coding. First we present a basic payment scheme to provide

incentives for each intermediate node to follow XOR protocol. We show that under our basic scheme, deviating from the XOR protocol leads to utility decrease in the experiments, thus all intermediate nodes have incentive to follow the XOR protocol. However, due to the nature of XOR network coding, the real cost can be reduced through XOR-ing multiple packets into one. To fight against overpayment problem, we propose an enhanced payment scheme using SVM model. In this scheme, sources share their historical data before transmitting, and these data are used to train a SVM for payment prediction. Payment prediction are categorized into different percentages of the maximum payment that we have in the basic scheme. The results show that the enhanced scheme can reduce the payment by up to 70%, while ensuring a low prediction error rate.

Algorithm 1 Certificate Processing

Input: number of nodes n, threshold t, a random number k, a sufficient long array of certification messages $M_1, M_2, \ldots$, a sufficient long array of primitive elements $\{g_1, g_2, \ldots\}$, a polynomial $P(x)$ with random efficient $\{c_0, c_1, ..., c_{t-1}\}$ and $c_0 \neq 0$.
Init:

1. The central bank creates secret shadow for each node:
$P(x) = c_0 + c_1 x + c_2 x^2 +, \ldots, + c_{t-1} x^{t-1}$,
$k_1 = P(1)$,
$k_2 = P(2)$,
$\ldots$,
$k_n = P(n)$

2. (k_i, i) is sent to node i

3. The central bank publishes the information for threshold decision sessions
$C_1 = (g_1^k, M_1 g_1^{c_0 k}), C_2 = (g_2^k, M_2 g_2^{c_0 k}), \ldots$

Repeat:

1. The central bank collects a threshold decision session result $\{i, d, M_h\}$ from node i, where d indicates the sequence of transmission session, h indicates the sequence of threshold decision session, and M_h is the certification message generated for hth threshold decision session.

2. The central bank checks the validity of M_h and determines whether it has been used before.

3. If no cheating message is found in $\{i, d, M_h\}$, the central bank accepts the payment recalculation request for node i's dth transmission session. And the central bank will punish the source node.

4. Else, the central bank punishes the node that sent the threshold session result.

5. The central bank repeats steps 1-4 if available.

Algorithm 2 Threshold Decision

Input: number of nodes n, threshold t, transmission session sequence d, threshold decision session sequence h, each node's own shadow information (k_i, i), public information for each threshold decision session $C_1 = (g_1^k, M_1 g_1^{c_0 k}), \ldots$, and a timer T.
Procedure:

1. Node i finds its payment miscalculated and thus broadcasts its threshold decision request, which contains his own calculation result $p(i, d)$, where i is the node sequence and d indicates it is the dth transmission session. Node i also starts the timer T.

2. Any other node that receives the message $p(i, d)$ from node i should calculate the corresponding payment for node i's dth transmission based on the loss probability matrix it holds.

3. If node j agrees with node i's claimed payment, it replies to node i with the information $(j, g_h^{kk_j})$, where h indicates it is the hth threshold decision session, and $g_h^{kk_j}$ is the modified secret shadow node j can share with node i.

4. If node j does not agree with node i, it does not response.

5. Once node i collects at least t shadows including its own, it stops the timer T. Without loss of generality, define those secret shadows by $(1, g_h^{kk_1}), ..., (t, g_h^{kk_t})$, it computes a_i following the equation $a_i = \prod_{s=1, s\neq i}^{t} \frac{(0-i)}{(i-s)} \ (module\ p)$. Then node i has $g'_h(i) = (g_h^{kk_i})^{a_i}$

6. From equation
$M_h g_h^{c_0 k} \prod_{i=1}^{t} g'_h(i) = M_h g_h^{c_0 k} g_h^{-c_0 k} = M_h$,
node i retrieves the certification message M_h

7. If timer T timeout and only less than t shadows have been collected, node i will end its threshold decision session.

Output: Node i reports the threshold decision session result $\{i, d, M_h\}$ to the central bank if it successfully recovers the certification message M_h and the central bank is available.

Algorithm 3 Source Node Payment Calculation

Init: total length of session data L, a set of all nodes, system parameter β

1. S computes its routing path and obtains a set of intermediate nodes $I = \{1, \ldots, n\}$
2. S calculates its payment to node i: $p_i^s(L) = \beta L$.
3. If S receives a feedback report from node i, then it pays credits p_{i-1}^s to node $(i-1)$.

Algorithm 4 Enhanced Payment Scheme for Source Node S

Init: length of session data L, a set of all nodes, system parameter β

1. S shares historical data and session information with other sources.
2. S computes its routing path and obtains a set of intermediate nodes $I = \{1, \ldots, n\}$
3. S calculates its maximum payment for this session: $p_{max}^s = \beta n L$.
4. S builds a training data set from historical information of all nodes.
5. S builds its test input x_s given path information of all concurrent sessions.
6. S trains the SVM model using the data set.
7. S inputs x_s to the SVM model and gets the decision z_s.
8. S computes its new overall payment p_{svm}^s as in Equation (2.21).
9. S pays its new payment to node $i \in I$ as in Equation (2.22), if S receives a feedback report from node $(i+1)$.

Chapter 3

Incentive Compatible Scheme for Cooperative Relay in Cognitive Radio Networks

3.1 Introduction

Cognitive radio networks [62][63][64] have received a lot of attention in recent years, because they allow secondary users to detect the spectrum not used by primary users and thus improves the utilization of spectrum. In cognitive radio networks, *cooperatively relay* [65]-[71] is a new approach in which nodes help each other to relay traffic, so that better performance can be achieved.

Two types of cooperative relay have been proposed in cognitive radio networks these years. The first type of cooperative relay is between primary and secondary users, where secondary users seek opportunities to relay data for

the primary user in exchange for some time for its own data transmission in primary users free spectrum. This type of cooperative relay [65][66],[81]-[89] has been extensively studied recently; lots of interesting results have been obtained, including some based on game theory. Although this kind of relay is a promising method to achieve better efficiency of the spectrum resource, works in this category generally assume the primary user is a licensed user who owns the spectrum property and may choose to lease portions of the spectrum to the secondary users. Thus they might not be suitable for other spectrum sharing models.

The second kind of cooperative relay is proposed to improve the spectrum utilization when primary users are not necessarily involved. In particular, Jia, Zhang and Zhang [67][68] notice that the spectrum availability of secondary users in cognitive radio networks is heterogeneous, and secondary users may have different traffic demands. Consequently, they propose protocols in which secondary users use their spare spectrum to relay traffic for other secondary users, in order to improve the spectrum usage of all secondary users. Thereafter, many studies have been conducted on this type of cooperative relay [69][70][71].

3.2 Towards Cheat Proof Cooperative Relay for Cognitive Radio Networks

3.2.1 Background and Motivation

In cognitive radio networks, J. Zhang and Q. Zhang [82] propose a protocol for cognitive radio networks in which secondary users relay packets for their primary user as rewards (in addition to paying the primary user) for allowing them to use the primary user's licensed frequency band. To analyze their protocol, they assume involved users are *selfish*, and model the interactions among secondary users as a strategic game, which is a part of an extensive game (more precisely, a Stackelberg game) that represents the entire process of cooperative relay. They elegantly show that the primary user can maximize its own utility while all secondary users reach the unique Nash Equilibrium (NE) in their strategic game.

While the existing protocols for cooperative relay are very interesting and useful, there is a crucial problem that has not been investigated: in reality, selfish users may *cheat* in cooperative relay, e.g. reporting false transmission rates about their relay channels, in order to benefit themselves. Such cheating behavior may harm other users and thus lead to poor system throughput. For example, in the Zhang-Zhang protocol [82], when secondary users are in the NE, each secondary user's equilibrium strategy depends on other users' secondary links' transmission rates. Consequently, if a selfish secondary user cheats by reporting a wrong transmission rate of its own secondary link, then other users may be misled to choose strategies that benefit the cheater and harm themselves. We present a detailed study of such cheating behavior, demonstrating how a user's cheating behavior can benefit himself and harm other users. We also illustrate how cheating behaviors affect the system throughput negatively.

Given the threat of selfish users' cheating, our objective in this work is to

suppress the cheating behavior of selfish users in cooperative relay, so that all these users have incentives to follow the protocol. We achieve this objective through two steps. In our first step, we focus on the interactions among secondary users, and design a basic scheme that gives selfish users incentives to follow the protocol faithfully, i.e., not to cheat. Our basic scheme guarantees that if a secondary user cheats in these interactions, the cheating behavior never benefits himself. Hence, under the basic scheme we design, secondary users have no incentive to cheat. In our second step, we use simple security techniques to extend our scheme to the entire process of cooperative relay, which involves not only the secondary users but also the primary user. The extended scheme suppresses cheating behavior throughout the entire process of cooperative relay, so that selfish users have incentives to follow the protocol faithfully all the way through the entire process.

Our contributions can be summarized as follows:

- We design a basic scheme for interactions among secondary users, which is the *first* cheat-proof scheme for cooperative relay in cognitive radio networks. In the model of strategic game, we rigorously prove that under our scheme, it is a dominant strategy for secondary users to faithfully follow the protocol. In other words, cheating is never beneficial under our basic scheme.

- We also extend our scheme to the entire cooperative relay process. In an extensive game model that involves all users, we prove that it is a Subgame Perfect Nash Equilibrium (SPNE) for both primary user and secondary users to follow our extended scheme.

- We consider fairness and propose an approach to reduce starvation of secondary users while maintaining good throughput.
- We perform extensive simulations. Results demonstrate that, without our schemes, a secondary user can cheat to benefit himself while harming other users. In contrast, with our schemes, a user's cheating is never beneficial to himself. By suppressing cheating behavior, our schemes improve the system throughput in face of selfish users.

3.2.2 Basic Scheme to Suppress Cheating

In the strategic game among secondary users, the set of players is S. Since we can easily detect the cheating behavior if a secondary user pays an amount not identical to what it claims, we assume the secondary users are smart enough that they always pay amounts identical to what they claim. For each $i \in S$, the action defined in our game is to report a payment c_i to the primary user. Based on the profile of all players' actions, player i gets utility

$$u_i = w(1-\alpha)R_i t_i - c_i, \tag{3.1}$$

where w is the amount of equivalent payment for each unit of data transmission rate, and t_i is ratio of the assigned access time to the duration of the third phase. Intuitively, this utility is equal to the benefit of accessing the free channel minus the payment to the primary user.

Given the system model, we can now build a cheat-proof scheme for cooperative relay. To illustrate the need for cheating suppression, we first briefly describe possible cheating behavior in (this model of) cooperative relay. Then,

to suppress such cheating behavior, we design and analyze a basic cheat-proof scheme for cooperative relay.

Cheating Behavior

In other existing protocols, the secondary users have to report their own data rates R_i to other users. However, in practice a selfish secondary user may cheat on such information in order to benefit itself. Based on this observation, we have the following *general* result on cheating:

Lemma 1. *Consider a cooperative relay protocol in which, for $i \in S$, $c_i = \theta_i((R_j)_{j\in S})$, $t_i = \omega_i((c_j)_{j\in S})$, where $\theta_i()$ and $\omega_i()$ are piece-wise continuous functions (for all $i \in S$). Define $\eta_i()$ as $\eta_i((R_j)_{j\in S}) = \omega_i((\theta_j((R_{j'})_{j'\in S}))_{j\in S})$. If there exists $i \in S$ such that there is a segment $(R_L, R_H) \subset (0, +\infty)$ on which*

$$\exists R_{-i} \; s.t. \; \frac{\partial \eta_i}{\partial R_i}((R_j)_{j\in S}) \neq 0, \tag{3.2}$$

then following the protocol is not a NE.

Proof. We use the value of R_{-i} from Equation (3.2). By Equation (3.2) we know there exists $R_i^\dagger, R_i^\triangle \in (R_L, R_H)$ such that $\eta_i(R_i^\dagger, R_{-i}) \neq \eta_i(R_i^\triangle, R_{-i})$. Without loss of generality, assume $\eta_i(R_i^\dagger, R_{-i}) < \eta_i(R_i^\triangle, R_{-i})$. Now consider a game in which player i's transmission rate is $R_i^\dagger$. Since

$$\begin{aligned} u_i((c_i, R_i^\dagger), (c_{-i}, R_{-i})) &= w(1-\alpha)R_i\eta_i(R_i^\dagger, R_{-i}) - c_i \\ &< w(1-\alpha)R_i\eta_i(R_i^\triangle, R_{-i}) - c_i \\ &= u_i((c_i, R_i^\triangle), (c_{-i}, R_{-i})), \end{aligned}$$

player i will gain more utility if it reports a false transmission rate $R_i^\triangle$ rather

than $R_i^\dagger$. Therefore, truthfully following the protocol is not a NE. □

Lemma 1 tells us that a secondary user i can increase its assigned access time by reporting a false value of its own transmission rate, and thus increase its own utility.

Hence, a natural question arises: can we detect cheating by examining the consistency between R_i and c_i using the equation $c_i = \theta_i((R_j)_{j\in S})$? Unfortunately, the answer is no for existing protocol, because in general a secondary user i can find a pair (c_i', R_i') that increases t_i and further u_i but satisfies $c_i' = \theta_i(R_i', R_{-j})$.

Below is a numerical example in the Zhang-Zhang protocol [82], in which we want to show that secondary users may cheat intelligently in order to avoid being detected and increase utility. In their model, $c_i = w(1-\alpha)(k-1)[\sum_{j\in S}\frac{1}{R_j} - \frac{k-1}{R_i}](\sum_{j\in S}\frac{1}{R_j})^2$ and $t_i = \frac{c_i}{\sum_{j\in S} c_j}$. [1] From Equation (3), the utility of each secondary user i is $u_i = \frac{w(1-\alpha)c_i R_i}{\sum_{j\in S} c_j} - c_i$. Assume that $w = 2$, $\alpha = 0.5$, $k = 3$, $R_1 = 2.5579, R_2 = 2.8126, R_3 = 3.4240$. If all users faithfully report their channel information, the payment of each user should be $c_1 = 0.4759, c_2 = 0.6072, c_3 = 0.8426$ from the above equation. And the utilities for users are $u_1 = 0.1562, u_2 = 0.2796, u_3 = 0.6555$. Now consider a situation in which user 1 cheats: suppose it chooses to report its transmission rate as $R_1' = 5.0$ and keeps its payment consistent with the reported transmission rate $c_1' = 1.2461$. From above utility equation, user 1's utility becomes $u_1' = 1.3943$, which is greater than its original utility $u_1 = 0.1562$. This false report also changes the utilities of users 2 and 3: $u_2' = 0.5249$ and $u_3' = 0.0034$. In this case, user 1's cheating

[1] We adopt the payment function in the Zhang-Zhang protocol, because it is the only one that considers secondary users' payments in existing related work.

behavior benefits himself but harms user 3.

Design of Basic Scheme

In order to suppress cheating in cooperative relay, the main idea underlying our design of a cheat-proof scheme is that we should make sure a user's utility is *never* affected by any other user's reported transmission rate.

To achieve this objective, we examine the definition of utility, Equation (3.1). For a secondary user i, the payment c_i and the access time t_i calculated in existing protocols can be affected by other users' reported transmission rates. Hence, we design a new method of payment determination and time assignment such that c_i and t_i are not affected by any other user's reported transmission rate. And each secondary user does not have to report its own transmission rate to the primary user and other secondary users.

Of course, to make the scheme practical, there are additional requirements on the time assignment method. For example, the assignment of access time should be fair to all secondary users in S. Furthermore, when a secondary user increases (decreases, resp.) its payment to the primary user, its assigned amount of time should increase (decrease, resp.) accordingly.

A summary of our basic cheat-proof scheme is given as Algorithm 5. We choose to use a simple method of access time assignment that satisfies all the above requirements. Our method first divides the total amount of time for secondary access evenly among all the secondary users in S. Then, it reduces the amount of time assigned to user i based on user i's payment. More precisely, the amount of time assigned to user i is multiplied by $1 - \frac{c}{c_i}$, where c is a constant determined by the primary user.

The above method of access time assignment normally produces some left-

Algorithm 5 Basic Scheme for Cheating Suppression

Input: S—selected secondary user set; α—time slot parameter; R_i—transmission rate; c—constant selected by primary user; w—payment equivalent to one unit of transmission rate.
Before each time slot, secondary user $i \in S$ does the follows:

1. Calculate (and make) the payment to the primary user

$$c_i = \sqrt{\frac{w(1-\alpha)R_i c}{k}}, \tag{3.3}$$

where $k = |S|$.

2. In the third phase of the time slot, use the following ratio t_i to compute the allocated time for own access:

$$t_i = \frac{1}{k}(1 - \frac{c}{c_i}). \tag{3.4}$$

over of the access time. Given Equations (3.3) and (3.4), we get the leftover time ratio in each time slot

$$T_{left} = 1 - \sum_{i=1}^{k} t_i = 1 - \frac{1}{k}\sum_{i=1}^{k}(1 - \frac{\sqrt{kc}}{\sqrt{w(1-\alpha)R_i}}). \tag{3.5}$$

Using this equation which connects T_{left} to c, we can easily obtain that $\lim_{c\to 0} T_{left}(c) =$ 0, which means the leftover can be negligible if the system parameter c is sufficiently small. We will evaluate how c can affect the leftover and each secondary user's utility in a numerical example in the evaluation section. We can show that although c may be very small, secondary users can still have positive utilities which ensures that our scheme is feasible.

Following this algorithm, we guarantee that user i's assigned time is not affected by other users' reported transmission rates. Moreover, the allocated access time of each secondary user increases if they pay more to the primary user. Note that although the primary user does not need to know each secondary user's own data transmission rate R_i, R_i does affect the secondary user's payment thus further influencing the primary user's utility.

Theorem 2. *In our scheme, it is a dominant strategy equilibrium (DSE) that all players $i \in S$ follow the protocol faithfully.*

Proof. Assume that s_i^* is the pure strategy of player i that follows the protocol faithfully. That is, s_i^* assigns probability 1 to action c_i^* where $c_i^* = \sqrt{\frac{w(1-\alpha)R_i c}{k}}$ and $k = |S|$. Based on Equation (3)(9)(10), when player i uses strategy s_i^*, the utility of player i is

$$\begin{aligned} u_i(s_i^*, s_{-i}) = & \frac{w(1-\alpha)R_i}{k}(1 - \frac{c}{c_i^*}) - c_i^* \\ = & \frac{w(1-\alpha)R_i}{k}(1 - \frac{c}{\sqrt{\frac{w(1-\alpha)R_i c}{k}}}) \\ & - \sqrt{\frac{w(1-\alpha)R_i c}{k}} \\ = & \frac{w(1-\alpha)R_i}{k} - 2\sqrt{\frac{w(1-\alpha)R_i c}{k}}. \end{aligned} \tag{3.6}$$

In contrast, consider the situation in which player i uses a mixed strategy $s_i \neq s_i^*$. Suppose that $c_i^{(1)}$, $c_i^{(2)}$, …, $c_i^{(m)}$ are all the actions assigned positive probabilities by s_i, and that their assigned probabilities are p_1, p_2, …, p_m, respectively. Clearly, s_i assigns positive probability to at least one action that is not identical to c_i^*. When action $c_i^{(j)}$ is taken, player i's utility is

$$\begin{aligned} u_i(c_i^{(j)}, s_{-i}) = & \frac{w(1-\alpha)R_i}{k}(1 - \frac{c}{c_i^{(j)}}) - c_i^{(j)} \\ = & \frac{w(1-\alpha)R_i}{k} - (\frac{w(1-\alpha)R_i c}{k c_i^{(j)}} + c_i^{(j)}) \\ \leq & \frac{w(1-\alpha)R_i}{k} - 2\sqrt{\frac{w(1-\alpha)R_i c}{k}} \\ = & u_i(s_i^*, s_{-i}), \end{aligned} \tag{3.7}$$

where the last identity is due to Equation (3.6).

Therefore, for strategy s_i, we have

$$\begin{aligned} u_i(s_i, s_{-i}) = & \quad (1 - \sum_{j=1}^{m} p_j) u_i(s_i^*, s_{-i}) \\ & + \sum_{j=1}^{m} p_j u_i((c_i^{(j)}, r_i^{(j)}), s_{-i}) \\ \leq & \quad (1 - \sum_{j=1}^{m} p_j) u_i(s_i^*, s_{-i}) \\ & + \sum_{j=1}^{m} p_j u_i(s_i^*, s_{-i}) \\ = & \quad u_i(s_i^*, s_{-i}), \end{aligned} \tag{3.8}$$

where the inequality is due to Equation (3.7).

Consequently, s^* is a DSE. □

3.2.3 Extended Scheme

The previous section presents a basic scheme to suppress cheating in the interactions among secondary users. In this section, we extend the scheme to suppress cheating throughout the entire process of cooperative relay, which involves both the primary user and the secondary users. In other words, we need to take into consideration the primary user's selection of relay users, and also aim to suppress possible cheating during this selection. The prerequisite of our extended scheme is that channel reciprocity holds for a reasonably long period of time and that we need to achieve a high level of accuracy in channel measurements. In some practical scenarios, the assumption of channel reciprocity is not reasonable. We will discuss the approaches that could measure relay channel infor-

mation without the assumption of channel reciprocity at the end of this section. We assume that there are some fixed sending power levels $(PW_1, PW_2, \ldots, PW_n)$ secondary users may choose from, and such power levels are already known by the primary user. In practice, this means secondary users use types of devices known by the primary user. The channel gain amplitudes are assumed to be the same for different sending power levels.

Cheating Behavior

As we have mentioned in technical preliminaries, in the first phase of cooperative relay, the primary sender PS distributes data to secondary users in S, and in the second phase, the secondary users in S relay data to the primary receiver PR. The primary user chooses the set S which provides the largest utility. From Equations (1.4)(1.5)(3.9), we know that the utility of the primary user is affected by the reported channel gain h_{0i} and h_{i0} of each secondary user $i \in SU$.

Suppose there is a secondary user i which has no chance to be selected as a relay user if it truthfully reports its relay related information. However, by cheating in values of $|h_{0i}|$ or $|h_{i0}|$, secondary user i may mislead the primary user to select itself as a relay user. As a result, this user i can benefit from the cheating behavior, while on the other hand the primary user and the system may be harmed.

Design of Extended Scheme

We model the interactions among all users, including the primary user and the secondary users, in each slot as a two-stage extensive strategic game with perfect information. Specifically, the utility function of the primary user is equal to the value of the overall throughput through cooperative relay plus the sum

of the payments collected from selected secondary relay users:

$$u_0 = wR_P + \sum_{i \in S} c_i. \tag{3.9}$$

The utility function of a secondary user is given above in Equation (3.1).

Although in practice there might be multiple primary users, we would like to first present our model in the case where only one primary user is present. The reason is that if there are multiple primary users, not only the secondary users, but also the primary users have to bid for their demanding resources from a game theory point of view. As a result, the game will become a double auction instead of our current extensive game model. We found it very challenging if we integrate the model of multiple primary users into current one, and would like to study this new model in the future.

We consider a radio model, in which each involved device can freely determine which power level among $PW_1, PW_2, \ldots, PW_n$ is used to send signals, and each such device can receive signals from others at any receiving power level. Clearly, a major challenge for designing the extended scheme is to correctly measure $|h_{i0}|$ and $|h_{0i}|$ of selfish secondary user i by the primary user, because these selfish users may use power control and cheat when reporting the channel information. To address this challenge, we require secondary users to send test signals to the primary user. [2]

Specifically, we require secondary users to send test signals at their highest and lowest power levels, respectively. PR computes $|h_{i0}|$ using the strengths of the received test signals. Since the test signals are transmitted at two different

[2]Throughout our work, we assume such test signals must be sent by the secondary senders, *not by the secondary receivers*, because only secondary senders are involved in data relay.

power levels, there are two results for $|h_{i0}|$, based on the highest transmission power and the lowest transmission power, respectively. If these two results are (roughly) equal to each other, then secondary user i has not cheated. Otherwise, secondary user i has cheated and should be punished (e.g., be excluded from relay permanently). To measure $|h_{0i}|$, we can use channel reciprocity and let PS compute the channel gain using the strengths of received signals of PS. [3]

The underlying idea of the above design is very simple: If a secondary user cheats, it can only decrease the power level when it is supposed to transmit at the highest power level, and it can only increase the power level when it is supposed to transmit at the lowest power level. The former definitely decreases the measured $|h_{i0}|$, while the latter definitely increases the measured $|h_{i0}|$. There is no way for a cheating user to keep the two measured values of $|h_{i0}|$ equal to each other.

Consequently, the extended scheme works by first measuring $|h_{0i}|$ and $|h_{i0}|$ correctly as described above. After that, the primary user first excludes the cheating users (if any) from the relay candidates, and then searches [4] for a proper set of relay users that can maximize its utility based on the information of all honest secondary users. Finally, the payment due and the secondary users' access time are computed just as in the basic scheme.

The details of the extended scheme are presented in Algorithm 6.

In our scheme, we compare two measured values of a channel gain using a threshold ϵ. This ϵ determines whether two values are "equal" to each other.

[3] The computed channel gains would be more precise if the primary user could repeat the measurement and take the average of the results.

[4] In existing protocols, e.g. [82], exhaustive search is used to enumerate all the possible set S. Depending on the application, we can either use the same approach, or pursue a better search strategy. We do not discuss this issue here in more detail, because it is out of the scope of our work.

Algorithm 6 Extended Scheme

Input:
$PW = \{PW_1, PW_2, \ldots, PW_n\}$ ($PW_1 <, \ldots, < PW_n$); system parameter $\epsilon > 0$; SU; $SU' = \phi$; $SU^* = \phi$.

Cheating Detection:

1. Secondary user $i \in SU$ sends test signal at power level PW_n.
2. PR (PS, respectively) receives the signal. Let the strength of the received signal be $Q_{R,n}$ ($Q_{S,n}$).
3. Secondary user i sends test signal at power level PW_1.
4. PR (PS, respectively) receives the signal. Let the strength of the received signal be $Q_{R,1}$ ($Q_{S,1}$).
5. PR computes $h_{i0,1} = Q_{R,1}/PW_1$ and $h_{i0,n} = Q_{R,n}/PW_n$. If $|h_{i0,n} - h_{i0,1}| < \epsilon$, then PS sends $(h_{i0,n} + h_{i0,1})/2$ as the measured values of $|h_{i0}|$ to PR. Otherwise, cheating is detected and $SU' = SU' \cup \{i\}$.
6. PS computes $h_{0i,1} = Q_{S,1}/PW_1$ and $h_{0i,n} = Q_{S,n}/PW_n$. If $|h_{0i,n} - h_{0i,1}| < \epsilon$, then $(h_{0i,n} + h_{0i,1})/2$ is used as the measured values of $|h_{0i}|$. Otherwise, cheating is detected and $SU' = SU' \cup \{i\}$.

Decision on S:

7. The primary user obtains the set of honest users $SU^* = SU - SU'$, and punishes the secondary users in SU'.
8. The primary user searches for the cooperative relay set $S \subseteq SU^*$, that provides the primary user with the maximum utility.
9. Each secondary user $i \in S$ computes c_i and t_i as in the Basic Scheme.
10. i pays c_i to the primary user.

Specifically, when $|x_1 - x_2| < \epsilon$, we say x_1 and x_2 are equal. To determine the value of ϵ, one possibility is to consider a slow fading model, in which the received power levels have log-normal distributions [36]. Given a signal sent at power level PW_i, the probability density function of the received power level is $f_i(x|PW_i) \sim lnN(\mu_i, \sigma_i)$. If we assume that the expected channel gain amplitudes are the same for different sending signal power levels, then we can compute the system parameter: $\epsilon = |(x_i/PW_1 - e^{\mu_1 + \sigma_1^2/2}/PW_1) + (e^{\mu_n + \sigma_n^2/2}/PW_n - x_j/PW_n)| = |x_i/PW_1 - x_j/PW_n|$, where x_i and x_j are receiving power levels of test sending signal PW_1 and PW_n respectively.

It is noteworthy that, in this algorithm, the decision on S is computed in a centralized manner. There are a couple of reasons that make it difficult to do the same thing in a fully distributed manner. However, it is still possible to do the same thing in a slightly more distributed manner. Detailed discussions of these issues can be found in next section.

Theorem 3. *It is a Subgame Perfect Nash Equilibrium that all users truthfully follow our schemes.*

Proof. There are only two stages in this extensive game: in the first stage the secondary users simultaneously report their channel information, and in the next stage the primary user determines relay users based on the information collected. Thus the terminal history of this game is in the form of $\lambda = ((\perp a_1^1 a_2^1 \dots a_n^1)(a_0^2 \perp \dots \perp))$, where $\perp$ means no action in current stage.

Suppose s^* is the strategy profile that all users truthfully follow our schemes, and $s_i^*|_\lambda$ is the strategy that user i induces from s^* in the subgame after history λ. For any $i \neq 0$ (secondary user), user i only acts in the first stage of the extensive game when history $\lambda = \phi$. Thus the only subgame that these players participate in is the entire extensive game itself. The strategy $s_i^* = (h_{0i}^*, h_{i0}^*, c_i^*)$, where h_{0i}^*, h_{i0}^*, c_i^* are secondary user i's *real* channel and payment information.

Suppose the player $i \neq 0$ picks a strategy $s_i' \neq s_i^*$, where $s_i' = (h_{0i}', h_{i0}', c_i')$, then $(h_{0i}', h_{i0}', c_i') \neq (h_{0i}^*, h_{i0}^*, c_i^*)$. Other players follow strategies s_{-i}^*. Now we consider two situations:

1. $h_{0i}' \neq h_{0i}^*$ or $h_{i0}' \neq h_{i0}^*$. By following Algorithm 2, player 0 can detect the cheating behavior and exclude player i from set SU before it determines the relay set S. Thus player i will finally have its utility $u_i(s_{-i}^*, s_i') = 0 \leq$

$u_i(s^*_{-i}, s^*_i)$.

2. $h'_{0i} = h^*_{0i}$ and $h'_{i0} = h^*_{i0}$, however $c_i \neq c^*_i$. If $i \in S$ (player i is chosen as a relay user), then similar to the proof of Theorem 4 (Section 3.3), we can show that $u_i(s^*_{-i}, s'_i) \leq u_i(s^*_{-i}, s^*_i)$. If $i \notin S$, then i is not chosen as a relay user, so simply we have $u_i(s^*_{-i}, s'_i) = 0 \leq u_i(s^*_{-i}, s^*_i)$.

Replacing s'_i with s_i in the above, we get that, for s_i of player $i \neq 0$ in this extensive game, we have

$$u_i(s^*_{-i}, s_i) \leq u_i(s^*_{-i}, s^*_i). \tag{3.10}$$

For the primary user, the utility function is $u_0 = wR_P + \sum_{i \in S} c_i$. Note that for $\forall i \in SU$, u_0 is not affected by transmission rate R_i. The primary user can obtain a new set $SU^* = SU - SU'$ where SU' is the set of all users detected cheating, and its strategy s^*_0 is to exhaustively compute all the possible utilities based on the information of users in SU^*, and finally determine the set S which provides the maximum utility. Clearly by Step 8, in subgame $\Gamma(\lambda^*)$ where all secondary users truthfully report their information in history λ^*, for $\forall s'_0 \neq s^*_0$ we have

$$u_0(s^*_{-0}|_{\lambda^*}, s'_0|_{\lambda^*}) \leq u_0(s^*_{-0}|_{\lambda^*}, s^*_0|_{\lambda^*}). \tag{3.11}$$

In any other subgame $\Gamma(\lambda')$ where $|\lambda'| = 1$ and $\lambda' \neq \lambda^*$, there is at least one secondary user which does not truthfully follow our scheme. For any user $i \neq 0$ among those cheaters, cheating in transmission rate R_i does not affect the utility of the primary user as we mentioned before. So we only need to consider cheating in reporting h_{0i} and h_{i0} or making a payment $c'_i \neq c_i$. If it reports false

information of h_{0i} or h_{i0}, the primary user can detect the cheating behavior because a false h_{0i} or h_{i0} can be detected through Algorithm 2. Thus based on collected information of users in SU^*, by Step 8 we have $\forall s'_0|_{\lambda'} \neq s^*_0|_{\lambda'}$,

$$u_0(s^*_{-0}|_{\lambda'}, s'_0|_{\lambda'}) \leq u_0(s^*_{-0}|_{\lambda'}, s^*_0|_{\lambda'}). \tag{3.12}$$

If in λ' user $i \neq 0$ deviates by making payment $c'_i \neq c_i$, the algorithm of primary user chooses S such that $u_0(s^*_{-0}|_{\lambda'}, s'_0|_{\lambda'}) - c_i + c'_i \leq u_0(s^*_{-0}|_{\lambda'}, s^*_0|_{\lambda'}) - c_i + c'_i$ which implies

$$u_0(s^*_{-0}|_{\lambda'}, s'_0|_{\lambda'}) \leq u_0(s^*_{-0}|_{\lambda'}, s^*_0|_{\lambda'}). \tag{3.13}$$

Note that Equations (3.12) and (3.13) are similar to Equation (3.11), although they are for the situation with history $\lambda' \neq \lambda^\star$. If in λ' a user both cheats in reporting h_{0i} or h_{i0} and makes payment $c'_i \neq c_i$, or if more than one users cheat, we can combine the above analysis to show $u_0(s^*_{-0}|_{\lambda'}, s'_0|_{\lambda'}) \leq u_0(s^*_{-0}|_{\lambda'}, s^*_0|_{\lambda'})$. But due to the notational complexity, we skip it here.

Hence, based on Equation (3.10)(3.11)(3.12)(3.13), we can conclude that s^* is a Subgame Perfect Nash Equilibrium in our extensive game. □

3.2.4 Extended Scheme Analysis and Discussions

We analyze the overhead of the extended scheme and discuss its assumptions and distributed implementation.

Overhead Analysis

The primary user's main computational overhead is the search for the best S,

Table 3.1. Complexity analysis in Section 3.2

	Primary User	Each Secondary User
Computation	$O(2^{N-1}N)$	$O(1)$
Communication	$O(N)$	$O(1)$

which must cover the 2^N subsets of SU. For each candidate subset S, computing the corresponding utility of the primary user takes time of $O(|S|)$. Hence, the total computational overhead of the primary user is $O(\sum_{S \subseteq SU} |S|)$, i.e., $O(2^{N-1}N)$. While this overhead is exponential in N, because N is normally small, this overhead is reasonable in most cases. For example, when $N = 10$ (i.e., there are 10 secondary users available for cooperative relay), $2^{N-1}N = 5120$.

Each secondary user only needs to do 2 subtractions, 3 multiplications, 3 divisions, and 1 square root. Hence, its computational overhead is $O(1)$.

Assume that each payment involves transmissions of $T_{payment}$ messages from the payer to the payee and also $T'_{payment}$ messages from the payee to the payer (where $T_{payment}$ and $T'_{payment}$ are both small constants). The primary user needs to receive no more than $6N + |S|T_{payment}$ transmissions, and needs to make no more than $2N + |S|T'_{payment}$ transmissions. Hence, the primary user has a communication overhead of $O(N)$.

Each secondary user needs to receive no more than $2 + T'_{payment}$ transmissions and make no more than $2 + T_{payment}$ transmissions. Hence, each secondary user has a communication overhead of $O(1)$.

The complexity results are summarized in Table 3.1.

Discussions of Assumptions

Number of Primary Users: The extended scheme is based on the assumption that there is only one primary user. If there is more than one primary user, all

the primary users should work together to form a "primary user union." This primary user union can use our schemes to negotiate with the secondary users, as if the union is a single primary user. In this way, our schemes can be used without significant modification.

The remaining question is how the primary users share the responsibility of providing access time to the secondary users, and share the payment from the secondary users. While this is actually another topic not so closely related co-operative relay, we propose a concrete way to share both the responsibility and the payment: Whenever a primary user occupies a time-spectrum block, this primary user should also take all the responsibility and all the payment associated with this time-spectrum block. We believe this proposed way of sharing is natural, fair, and easy to use.

Distributed vs. Centralized Scheme: One may also be interested in whether our schemes should be considered distributed or centralized. We argue that our proposed schemes are distributed in the sense that they require communications between the primary and secondary users, that the computation of each c_i is done by secondary user i, and that access time t_i is computed by secondary user i and by the primary user simultaneously, so that both of them are aware of t_i. However, the decision on S in the extended scheme is made by the primary user in a centralized manner, once the needed information is collected.

Channel Reciprocity: The extended scheme we design in the main file is also based on the assumption of channel reciprocity, however this is not always true in practice. If the assumption of channel reciprocity does not hold, the problem becomes very challenging. Because we can only let the primary user send test signals and let the secondary users compute and report the channel gains of

the relay channel from the primary sender to the secondary users. In general, measuring channel characteristics between a pair of wireless devices may not be feasible when one of the devices is cheating and the other one obtains no extra information. If a secondary user cheats in reporting a false channel gain in this new model, the primary user itself is not able to detect the cheating behavior in current solution. And obviously our approach of measurements cannot be used directly any more or adapted easily to address the problem as long as no one else monitors the received power level of test signals.

One possible approach to sidestep this problem is as follows: First we assume that the primary user knows the weakest receiving power level of each secondary user. Then we let the primary user send test signals using decreasing power level. The test signal always contains an encoded index number. The weaker the sending power level is, the larger this index number will be. Each secondary user reports back all encoded numbers it receives. The primary user computes the channel gain of the relay channel based on secondary user's weakest receiving power level and the corresponding sending power level obtained from its largest reported index number. On the other hand, the largest reported index number is considered as the credits this secondary user obtains in this time slot. Each secondary user has to pay certain credits to be allowed to relay data in one time slot. This approach is designed based on the belief that selfish secondary users always have an incentive to gain more credits so will truthfully report weakest power levels they receive. The drawback of this approach is that it will be quite energy consuming if there are lots of available channels. And another credit system is introduced into the system.

The above approach based on the weakest receiving power level may not

work well if there is multipath fading. In particular, the computed channel characteristics may vary from time to time, and may be very inaccurate. For remedy, we distinguish two cases. First, in a benign case, we may repeat the process of channel characteristics computation a few times and use an algorithm to put together the computed values of channel characteristics. For example, the algorithm may first remove those clear outlier values and then take an average of the remaining values. We call this case "benign" because in this case the variation and inaccuracy caused by multipath fading are limited. Hopefully, these simple measures can remedy the limited variation and inaccuracy.

There is also a malevolent case, in which simple measures cannot work. In this worst situation, in order to sidestep the above difficulty, we may need to use a piece of tamper-proof hardware. This piece of hardware either controls the transmissions of test signals directly, or keeps an authentic record of the transmitted test signals so that the device can be audited in the future. With this piece of hardware installed, nodes will have to behave honestly when measuring the channel characteristics. Even if the measured values may still have errors, nodes have no choice except using these values. While this approach based on tamper-proof hardware is very powerful, its drawback is also very clear—the high cost of tamper-proof hardware, and the possible legal and social issues related to the requirement of installing tamper-proof hardware.

Discussions of Distributed Implementation

In the extended scheme we present in the main file, the decision on S is made in a centralized manner: Each secondary user sends test signals so that its channel gains can be computed by the primary user, and cheating in test signals can be detected. After removing the cheating secondary users, the primary user

uses the computed channel gains to figure out the set S that is best for itself. Hence, a natural question is whether we can implement this decision process in a distributed manner. We notice that it is very difficult to do so, because of two reasons:

First, the decision on S relies on detecting and removing secondary users who cheated in the process of measuring channel gains $|h_{i0}|$ and $|h_{0i}|$. Since h_{0i} and h_{i0} are for the channels between the primary user and the secondary user i, the decision of whether i is cheating and should be removed must involve the primary user. Since this applies to all secondary users, the detection and removal of cheating users is a centralized process that can be hardly done in a distributed manner.

Second, once the cheating secondary users have all been removed, the search for set S targets at maximizing the utility function of the primary user. Therefore, if we want to do this search in a distributed manner, we will need to require the primary user to distribute the information needed in this search, and then collect the results of the search back from the secondary users. Such a distributed search would have less efficiency and also less security.

Nevertheless, it is still possible to make the process a little more distributed than the centralized version of the extended scheme. We observe that, when secondary user i sends test signals, not only the primary user, but also other secondary users can receive these signals. Hence, if secondary user i cheats, other secondary users can also detect cheating. Clearly they can report the results of their detection to the primary user, so that the primary user can focus on computing the channel gains and leave the job of cheating user detection to secondary users.

The above proposed implementation is not fully distributed, but is just slightly more distributed than the fully centralized version. It main advantage is that we can improve the efficiency a little by requiring the secondary users to sequentially send test signals at their highest power level only, and the test signals at their lowest power level can be transmitted in parallel. The cost for this improvement in efficiency is that we need to deal with the additional security threat of secondary users falsely reporting detection of cheating.

Primary-Secondary User Communications

Our work has an implicit assumption that is inherited from the existing literature: the primary user can communicate with the secondary users. One may notice that, given this assumption, the primary and secondary users may be considered in the same network. Because there are many existing ways to optimize the performance of a wireless network, and because some of the existing ways may lead to better performance of the network than ours, one may wonder why our work is still of value given this assumption. Below we present our answer to this question.

First, before we explain the value of our work in detail, we stress that the above question actually applies not only to our work, but also to all works that involve secondary users leasing spectrum from the primary user. As mentioned in, such spectrum leasing requires communications between the primary and the secondary users, in order to determine various parameters and factors. Hence, the question we are answering is actually why such spectrum leasing is useful, given that the network consisting of the primary user and the secondary users may get better performance if some other techniques are used.

The fundamental reason is that, in the situation we study, the involved pri-

mary user and secondary users are all selfish. In comparison, the alternative optimization techniques that may lead to better performance are designed for situations without selfish behavior. Hence, if they are used in our situation, the selfish users may deviate from the protocol in order to pursue their own interests, preventing them from achieving the performance objective.

In fact, spectrum leasing is analogous to incentive-compatible routing in wireless ad hoc networks: For ad hoc networks, we also have very good routing protocols that do not consider selfish behavior, such as DSR. If there is no selfish behavior, such alternative routing protocols may lead to much better performance than any incentive-compatible routing protocol. However, because there can be selfish behavior in civilian applications of ad hoc networks, people recognize the need for incentive-compatible routing protocols and have designed a large number of them.

Therefore, while there exist methods to optimize the performance of the wireless network (which, in our case, consists of the primary user and the secondary users), in our situation, we still need an approach that can stimulate the users to cooperate in the whole process. Otherwise, those selfish secondary users would not provide their relay (or any other type of assistance) to the primary user. Spectrum leasing is studied, and this work stimulates secondary users to cooperate by giving them access to the spare spectrum of the primary user; furthermore, it stimulates the primary user to cooperate by allowing the primary user to collect payment in addition to getting the benefit of secondary users' relay services. Consequently, we believe it is a good attempt to achieve better performance in face of selfish behavior.

Discussions of Slow Fading Channel Recall we have an assumption that all

channels are slow fading channels. In the extended scheme, the use of a threshold value ϵ also depends on this assumption. Clearly, this assumption is suitable for the scenarios in which packets are exchanged frequently [4]. So naturally one may ask whether our extended scheme has any requirement in this aspect.

If we only look at the test signals transmitted to determine the channel gains and detect cheating, there are a good number of transmissions being made in a short amount of time. Hence, if we limit our attention to this part, we may think the assumption of slow fading channel is naturally valid for our situation and thus there is no need to introduce any additional requirement.

Nevertheless, if we take the transmitted data packets into account, we may get a different opinion. Without an additional requirement, the data packets might be transmitted infrequently, which implies that the assumption of slow fading channel might not be valid. Even if data packets are indeed frequently transmitted, when a sufficiently large amount of time has passed after the measurement of channel gains, the scheme may lose its incentive compatibility, because the current channel gains have become significantly different from the measured values.

Therefore, we need to introduce an additional requirement here, and this requirement is not just frequent transmissions of data packets. More precisely, we require that measurements of channel gains are performed repeatedly, and that once the channel gains have been measured, data packets are transmitted frequently. The cycle of channel measurement–data transmission should not be too long, so that the assumption of slow fading channel can be valid.

We realize that not all practical scenarios satisfy our requirement above.

[4] The authors thank an anonymous reviewer for pointing this out.

Hence, our scheme is not always applicable. It should be only applied to those scenarios that satisfy our requirement.

3.2.5 Fairness

In this section, we study the fairness issue of our schemes. In some scenarios under existing cooperative relay protocols, the primary user may always select certain secondary users with higher relay transmission rates as its cooperative relay candidates. Consequently, the primary user's free channel is accessed by only a few users. Other secondary users have no opportunity at all to access the free channel because of their low relay transmission rates. Thus we need to consider how to allocate resources to all possible relay candidates wisely.

One may suggest that a scheme is fair only if all secondary users obtain roughly equal shares of the spectrum. However, we argue that this may not be a good fairness metric for our schemes. Recall in our system model, the secondary users share the primary user's free channel if they are selected as relay nodes. And the achievable throughput of each selected user i is $R_i t_i$, which is determined by the transmission rate R_i. From Equations (3.3) and (3.4), we know that the two users i and j have the same achievable throughput $R_i t_i = R_j t_j$, if and only if $R_i = R_j$. On the other hand, the secondary users' transmission rates are predetermined and might be different. Therefore, it is not appropriate for the primary user to equally share the spectrum with all secondary users in our system.

Throughput Ratio Now let us take a closer look at the throughput ratio between different secondary users and examine how large it could be. Assume that all

secondary users follow our schemes (because, as we have shown, they have incentives to do so). Then using Equations (3.3) and (3.4), we can easily obtain that the throughput ratio between the two users i and j is

$$\begin{aligned}\eta_{ij} &= \frac{R_i t_i}{R_j t_j} = \frac{R_i(1-\frac{c}{c_i})}{R_j(1-\frac{c}{c_j})} \\ &= \frac{R_i(1-\frac{\sqrt{ck}}{\sqrt{w(1-\alpha)R_i}})}{R_j(1-\frac{\sqrt{ck}}{\sqrt{w(1-\alpha)R_j}})} \\ &= \frac{R_i-\frac{\sqrt{ckR_i}}{\sqrt{w(1-\alpha)}}}{R_j-\frac{\sqrt{ckR_j}}{\sqrt{w(1-\alpha)}}}.\end{aligned}$$

If all secondary users' transmission rates are sufficiently high (i.e., if for all $i \in S$, $R_i >> \frac{ck}{w(1-\alpha)}$), then we can easily get that

$$\eta_{ij} \approx \frac{R_i}{R_j} \leq \frac{\max_\ell R_\ell}{\min_\ell R_\ell}.$$

Therefore, $\max_\ell R_\ell / \min_\ell R_\ell$ is an upper bound for η_{ij}. Nevertheless, if the above condition is not satisfied, then η_{ij} might become large. For example, consider the extreme case in which both R_i and R_j are close to $\frac{ck}{w(1-\alpha)}$. In this case, we get that

$$\eta_{ij} \approx \frac{\sqrt{R_i}-\frac{\sqrt{ck}}{\sqrt{w(1-\alpha)}}}{\sqrt{R_j}-\frac{\sqrt{ck}}{\sqrt{w(1-\alpha)}}},$$

which implies that a small difference between the transmission rates R_i and R_j can lead to a large difference between the throughputs achieved by the two secondary users i and j. Consequently, in order to have good fairness, the primary user should make sure an appropriate value is chosen for c.

Starvation We study the fairness by examining starvation. A secondary user is considered to be starving, if it has never been selected as a relay candidate during the cooperative relay process. To achieve fairness, we focus on the starvation percentage of all secondary users and propose a possible approach to balance the tradeoff between starvation percentage and scheme effectiveness.

In Algorithm 2, the primary user always chooses the set S which brings with it the maximum utility. In order to reduce starvation, we propose that the primary user should consider all the possible sets of relay users as long as the expected utility of primary user is greater than $\tau(0 < \tau \leq 1)$ of its maximum utility, where τ is a system parameter. Given all available sets that satisfy, the primary user determines S according to the following priorities: (1) it first searches for the set with the largest number of secondary users that have never been selected in the past; (2) if there is more than one set available from the first step, then it chooses the set with the largest number of secondary users. If there is more than one set available after step (2), then randomly pick one set. When the above approach is applied to the extended schemes, we get a starvation-reduced scheme. Note that just like the extended scheme, our starvation-reduced scheme is also based on the basic scheme, and thus can also prevent secondary users from cheating.

Besides the above analysis and discussion, we also empirically study fairness. Specifically, we measure the starvation percentage of our extended scheme and that of the starvation-reduced scheme in the Evaluation section. In addition, although we do not aim to equally share the primary user's free spectrum with secondary users in our system, we measure the Jain's fairness index [35] and make comparisons. The results can be found in the evaluation section.

In summary, we have the following three contributions in terms of fairness analysis: First, we provide a throughput ratio analysis, in which we derive an upper bound for throughput ratio under a condition and also discuss the situation not satisfying this condition. Second, we analyze the starvation of secondary users and propose a starvation-reduced scheme. Third, we empirically study Jain's fairness index, the results of which can be found in the fairness analysis section.

3.2.6 Fault Tolerance, Security and Incentive Issues

When our schemes are used in practice, some nodes may malfunction or even fail. In addition, security attacks may be launched by selfish or malicious nodes. Although the focus of this work is on incentives rather than security, for practical purposes, we still consider fault tolerance and security issues and discuss the measures we need to take against them.

Fault Tolerance To defend our schemes against malfunctioning nodes, the simplest approach might be to require nodes to repeat their actions a few times. For example, in the basic scheme, the value c_i can be transmitted more than once, so that the primary user can use the consistency of received values to detect abnormal behavior of the nodes. In the rare case that a node is malfunctioning but its transmitted values are always consistent, our measure against false value submission attack can be used (please see below).

To detect possible node failure, the primary user should monitor all the involved secondary users' actions. If a node appears to have not taken any action for a certain amount of time, the primary user should send a hello message to

that node, to make sure it is still there. If there is no response to the hello message within a reasonable amount of time, then the node should be marked as failed and thus removed from all future considerations. If a failed node comes back in the future, it should authenticate itself to the primary user and request the primary user to reset its state to active.

Security Issues The very first security issue to consider is false value submission attack we mentioned above. Specifically, when our scheme requires a corrupted node to transmit a certain value to the primary user, this node can intentionally submit a wrong value in order to mislead the primary user. To mitigate such attacks, the primary user can use an outlier detection algorithm to examine all the submitted values, and remove those values that are clearly unreasonable. Of course, this measure cannot completely prevent false value submission attacks, because a corrupted node may just submit a slightly modified value, rather than a greatly changed one. A better measure to deal with false value submission attack needs to be studied in the future work.

Another security issue to consider is masquerade attack. A secondary user may masquerade as another secondary user, and send false channel information to mislead other users. For example, a secondary user has no chance to be selected as a relay user in the first place because of its low transmission rate on relay channel. However, it masquerades as some of its neighbors and sends false channel information of neighbors. Upon receiving these false information, the primary user mistakenly believes that the cheater has a higher transmission rate on the relay channel. As a result, the cheater is selected and thus benefits from its cheating behavior. To defend against such an attack, we can use the digital signature method: Each secondary user has to create a pair of keys, one

public and one private, and publish the public key before the protocol begins. Whenever a secondary user reports its information, it should sign the message using its private key. Each receiver should authenticate the sender by verifying the message using the public key. As long as the private key is kept confidential, another user cannot masquerade as this user under this defense method. For better security, each user could change the key pair periodically. Similarly, this mechanism can also defend against data corruption, in which a secondary user may tamper with the payload of its relayed packets during transmission.

Yet another possible security issue is payment fraud. Since we did not consider this type of attack in our game model, we need to discuss it here to complete our analysis. A secondary user may claim to have paid a certain amount of credits while actually paying nothing. The primary user may be misled by such attack and as a result obtain less utility than it expected. For example, a secondary user claims it will pay $10 if it is selected by the primary user, and the primary user really selects this cheating user as relay candidate. However, after the relay session, the secondary user pays less than $10 or does not pay at all. The primary user may suffer from this type of attack. To defend against payment fraud, each secondary user is required to pre-pay a certain amount of payment before each transmission session. Once the primary user receives the claimed payment from each user, the claimed amount of payment should be transferred from secondary user's account to the primary user's account. The feasibility of this method is based on the common assumption that in case of payment fraud the secondary user will always pay less than it claims. Under such defense method, payment fraud will no longer work, because the credits are transferred as soon as the payment is claimed.

Denial of service (DoS) may also be an attack conducted by some malicious users, in order to reduce the system performance. Here by default DoS attack means the attack conducted above (also including) MAC layer, which aims to consume the resources of a target. Sending powerful signals in order to interfere with other signals is not considered as a DoS attack in this work. For example, a secondary user may spam the payment information to the primary user while reporting, or broadcast meaningless packets while other users relay data. However, our system is by nature immune to secondary users' DoS attacks, due to the communication mechanism between the primary user and the secondary users: First, each secondary user only needs to report its payment to the primary user once. Report spamming is easy to detect. Second, when secondary users help relay data for the primary user, each relay secondary user only communicates with the primary sender and primary receiver. One can simply discard any received packet from other secondary users. Third, when a secondary user accesses its allocated spectrum, it may also discard any incoming packet from others. Therefore, in the entire process of our schemes, no users need deploy additional resources for unexpected senders because all unexpected information can be simply discarded.

Incentive Issues This work focuses on the incentives of nodes in the process of cooperative relay. However, our study has covered one aspect of node incentives in this process, namely the "negotiation" between the primary and secondary users to determine what nodes should participate the cooperative relay, how much access time the participants receive as compensation, and how much they should pay the primary user. When the cooperative relay actually starts, there will be another sort of incentives issue, namely how to make sure the par-

ticipating nodes really relay the data as they promised.

Since there are only two hops for the transmission of any packet in cooperative relay, PR clearly knows how many packets have been transmitted by any relay node (assuming the communication links are reliable). Comparing the actual transmissions with the promised transmissions, PR can easily decide whether each relay node has kept its promise. The only type of cheating PR needs to worry about is that relay nodes might transmit packets that do not originate from PS. (For example, a relay node may skip listening to PS, and directly transmit a number of randomly generated packets to PR, in order to get the access time from the primary user.) To deal with this type of cheating, PS needs to attach a MAC to each transmitted packet, and PR needs to check the MAC of all received packets.

3.2.7 Evaluations

We implement our schemes and conduct extensive experiments using GloMoSim [37]. First, we measure and compare the utilities of secondary users when a cheat-proof scheme is absent and present respectively. Second, we measure the system throughput and observe how it is affected by the cheating behaviors. Third, we evaluate the starvation percentage of secondary users when our proposed approach for starvation reduction is used and not used respectively, and we also evaluate the fairness following Jain's fairness index and compare the results. Finally we measure the payments of secondary users in random cases when our algorithms are implemented.

User Utilities

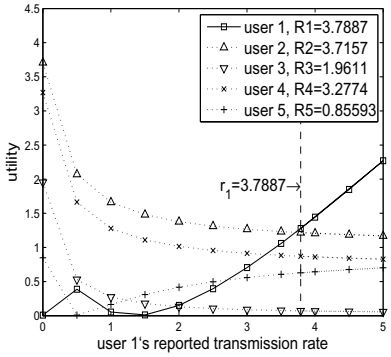

Figure 3.1. Utilities when user 1 cheats—no cheat-proof scheme in Section 3.2.

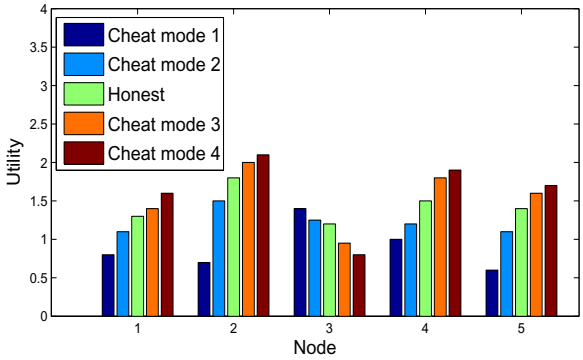

Figure 3.2. Utility changes when user x cheats in reporting R_x—no cheat-proof scheme in Section 3.2.

To see how our cheat-proof scheme affects user utilities in cooperative relay, we measure selfish secondary users' utilities in two scenarios of cooperative relay. In the first scenario, the Zhang-Zhang protocol [82] is executed without any cheat-proof scheme, while in the second scenario, our cheat-proof scheme is used. In all these experiment, $|S| = k = 5$ and we use 1, 2, 3, 4, 5 to denote the 5 users in S. Other parameters used in calculating utilities are $w = 2$, $\alpha = 0.5$ and $c = 0.01$.

User Utilities When No Cheat-Proof Scheme

As mentioned above, the first scenario in our evaluations is that the Zhang-

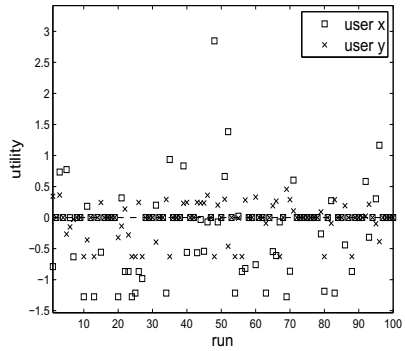

Figure 3.3. Utility changes when user x cheats in reporting h_{0x}, h_{x0}, R_x, c_x—no cheat-proof scheme in Section 3.2.

Zhang protocol is executed without any cheat-proof scheme. The three sets of experiments are performed as follows.

In the first set of experiments of this scenario, we measure utilities when user 1 cheats but other users are honest. The transmission rates of secondary users in this set of experiments are: $R_1 = 3.7887, R_2 = 3.7157, R_3 = 1.9611, R_4 = 3.2774, R_5 = 0.85593$. We let user 1's reported transmission rate r_1 vary from 0.0 to 5.0, while keeping its payment c_1 consistent with r_1, i.e., $c_1 = w(1-\alpha)(k-1)[\sum_{j\in S}^{j\neq 1} \frac{1}{R_j} - \frac{k-2}{r_1}](\sum_{j\in S}^{j\neq 1} \frac{1}{R_j} + \frac{1}{r_1})^2$.

Figure 3.1 shows five secondary users' utilities for different values of r_1. Clearly, user 1 can cheat to benefit himself. For example, if user 1 reports $r_1 = 4.5$ which is higher than its actual transmission rate $R_1 = 3.7887$, its utility increases from 1.2765 to 1.8479. However, such cheating behavior may harm other users. For example, when user 1 reports $r_1 = 4.5$, the utility of user 2 decreases from 1.2166 to 1.1861.

In the second set of experiments, we consider a total of five ($|S| = 5$) randomly picked secondary users. For each secondary user, we measure its utility

under five different strategies: $\delta = -40\%, -20\%, 0, 20\%, 40\%$, where δ means the difference between the claimed transmission report and the real one. Other users behave randomly when we measure one secondary user. The payment of each secondary user is calculated and reported correspondingly based on its *claimed* transmission rate

In Figure 3.2, the utilities are measured when there is no cheat-proof scheme. From the result, we notice that some users benefit when they report a smaller transmission rate, while the others benefit when they report a greater one. In this set of experiments, no user has obtained the greatest utility when it truthfully reports its transmission rate. Therefore, a secondary user may obtain greater utility if it properly chooses a false report rather than the real one, and thus has incentives to cheat.

The third set of experiments of this scenario consists of 100 runs and in each run the primary user follows Algorithm 2 to determine the S out of SU ($|SU| = 20$). Also a randomly selected user x uses a different cheating strategy: It reports random channel gains h'_{0i}, h'_{i0} ($-80dB < |h'_{0i}|, |h'_{i0}| \leq -50dB$), a random transmission rate r_x ($0.0 \leq r_x \leq 5.0$), and pays a random amount c_x ($0.0 \leq c_x \leq 2.0$).

Figure 3.3 shows utility changes of x and another user y who is randomly picked in each run just like in the second set of experiments. In all the 100 runs, we can see that x's random cheating benefits himself in 15 runs, and harms y in 24 runs. This again verifies that, if a user chooses its cheating behavior smartly, the cheating behavior would have a good chance of benefiting himself.

The results of all the three sets of experiments in the first scenario clearly indicate that it is necessary to use a cheat-proof scheme in cooperative relay.

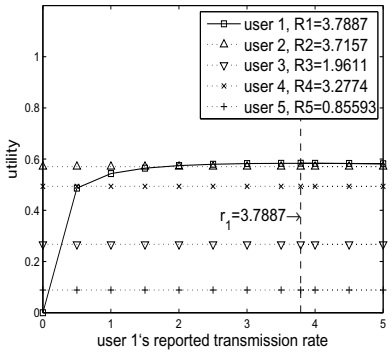

Figure 3.4. Utilities when user 1 cheats—our scheme in Section 3.2.

User Utilities When Our Scheme Is Used

In the second scenario, we repeat the three sets of experiments done in the first scenario. Recall that, in this scenario, our cheat-proof scheme is used to suppress cheating behavior. In the first two sets of experiments, we use our basic scheme, while in the third set of experiments, we use our extended scheme. For the purpose of comparison, we keep using the notation "reported transmission rates" in these evaluations as in the first scenario, although these transmission rates are not actually reported to the primary user in our scheme.

The results of the first two sets of experiments with the presence of our basic scheme are presented in Figures 3.4 and 3.5. In the first set of experiments, user 1's cheating behavior no longer benefits himself. As shown in Figure 3.4, user 1 can obtain its optimal utility only when it reports its transmission rate truthfully. Any cheating behavior will result in loss of its utility. Hence, user 1 has no incentive to cheat.

In the second set of experiments, we can also see from Figure 3.5 that each secondary user's cheating behaviors never benefit himself. The greatest utility is only obtained when each user honestly reports its transmission rate. Therefore,

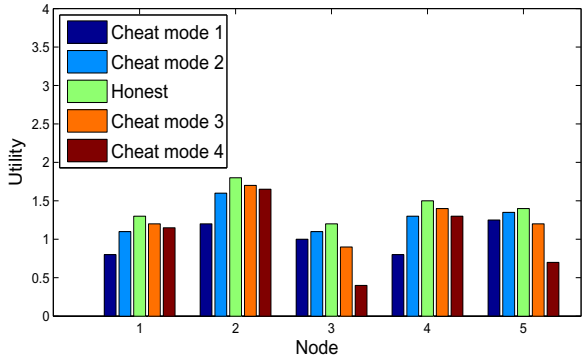

Figure 3.5. Utility changes when user x cheats in reporting R_x—our scheme in Section 3.2.

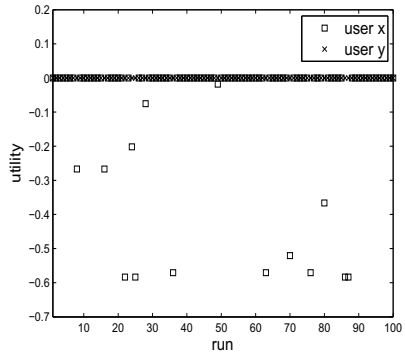

Figure 3.6. Utility changes when user x cheats in reporting h_{0x}, h_{x0}, c_x—our scheme in Section 3.2.

we have verified that each user has no incentive to cheat.

The results of the third set of experiments with the presence of our extended scheme are presented in Figure 3.6. Just as we see in the second set, user x's random cheating behavior never benefits himself. Furthermore, the cheating behavior no longer harms other users because of the presence of our extended scheme.

The results of all the three sets of experiments in the second scenario indicate that our cheat-proof scheme is effective in suppressing cheating behavior.

How Parameter c Affects Leftover Time and User Utilities

In each time slot under our algorithms, the primary user will have some leftover time which is not assigned to any secondary user after resource allocation. We have shown that the leftover time is affected by system parameter c. Furthermore, when the value of c approaches zero, the length of the leftover time ratio (the ratio of the leftover time to the duration of third phase in a time slot) approaches zero as well.

In this part, we set up experiments in which we observe how parameter c affects the value of the leftover time ratio in simulation. We assume $|S| = 5$, $w = 2$ and $\alpha = 0.5$. The network topology of these experiments is the same as in the first set of experiment data presented, and all five of the secondary users are honest. The transmission rates of these secondary users are the same as in the first experiment data: $R_1 = 3.7887$, $R_2 = 3.7157$, $R_3 = 1.9611$, $R_4 = 3.2774$, $R_5 = 0.85593$ which are randomly picked before simulation. The secondary users truthfully report their transmission rates to the primary user and the primary user computes the payment c_i and the assigned access time ratio t_i for each secondary user following the Algorithm 5. The value of system parameter c varies from 0 to 0.01, with step of 0.001 during the tests. In each test, we record the leftover time ratio $T_{left} = 1 - \sum_{i=1}^{5} t_i$. Thus we have 11 different results on the leftover time ratio T_{left} which is shown in Figure 3.7.

In Figure 3.7, we notice that when c is set to be 0, the leftover time ratio is 0 as well, which means there is no leftover time after resource allocation. This can be actually be derived from Equation (3.4) because when c equals 0, $t_1 = t_2 = \ldots = t_k$, i.e., the available access time is equally allocated to all selected secondary users, making our incentive schemes no longer useful. Hence zero

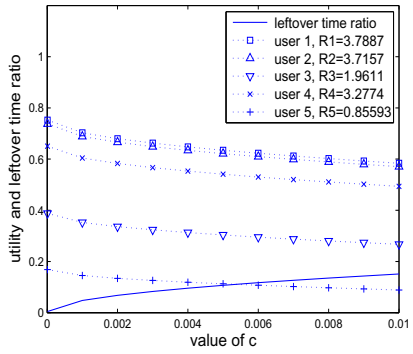

Figure 3.7. The leftover time ratio and secondary users' utilities in Section 3.2

is not an appropriate value of c in practice and we have to keep c positive. We can also observe the fact that T_{left} decreases when c decreases, which can also be theoretically derived from the definition of T_{left} in previous section.

Other than measuring the value of T_{left} in this part, we also measure the corresponding computed utility of each secondary user. Because the utility of each secondary user is affected by parameter c as well (from Equations (3.1)(3.3)(3.4)). We need to make sure the utilities are positive and the algorithm stays feasible when we change c.

To measure the secondary users' utilities, we use exactly the same experiment settings as above. The primary user computes t_i and c_i of each secondary user in a test run and we record the computed utilities of all five secondary users in all 11 runs. The results are shown in Figure 3.7. We can see that all utilities decrease but stay positive in all runs, which means our algorithm is still able to function well when parameter c has different values.

Now let us consider two example scenarios. The first example scenario is that $c = 0.01$, and the second is that $c = 0.001$. Below are our experimental results in these two example scenarios:

- In the first example scenario, the leftover time ratio is 0.1512. The secondary users' utilities are 0.5836, 0.5707, 0.2670, 0.4936, and 0.0884, respectively.

- In the second example scenario, the leftover time ratio is 0.0478. The secondary users' utilities are 0.7027, 0.6886, 0.3526, 0.6043, and 0.1450, respectively.

It is easy to see that in the second example scenario, the leftover time ratio is lower, which means less time is wasted. It is also easy to see that in the second example scenario, the secondary users have higher utilities. These benefits are all consequences of having a smaller (but still positive) value of c.

Throughput

In the previous subsection, we have seen that, without a cheat-proof scheme, cheating behavior may harm other users. In this subsection, we study how such cheating behavior of secondary users affects the system throughput. Specifically, we study a cognitive radio network with a primary user plus 20 secondary users, in which the Zhang-Zhang protocol [82] is running. The total system throughput can be calculated as $T = R_P(\alpha, \beta, S) + (1 - \alpha) \sum_{i \in S} R_i t_i$.

In the first experiment, we measure the total system throughput when there are different numbers of cheating users. In particular, we can verify that when there is 0 cheating user, the total system throughput achieved by the Zhang-Zhang protocol is maximized. We compare the system throughput when selfish behaviors are not suppressed in the Zhang-Zhang protocol with the system throughput when selfish behaviors are completely suppressed by our scheme. We would like to show that cheating behavior of secondary users will lead to

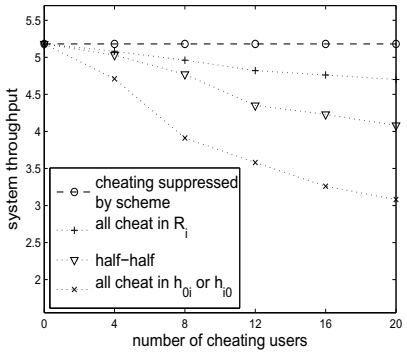

Figure 3.8. Total system throughput in Section 3.2.

a decrease in total system throughput. We also notice that the performance degradation caused by overhead in our scheme is almost the same as in the Zhang-Zhang protocol. Hence the system throughput achieved in our scheme, which can efficiently suppress cheating behaviors, is very close to the maximum system throughput of Zhang-Zhang protocol.

Figure 3.8 shows the measured throughput. Recall that a secondary user i may cheat in interacting with other secondary users (i.e., cheat in reporting its own transmission rate R_i), or in interacting with the primary user (i.e., cheat in reporting h_{0i} or h_{i0}). In this experiment, for each number of cheating users, we consider three possibilities: (1) Cheating in reporting R_i. In particular, a cheating user i always chooses random (c_i, r_i) $(r_i \neq R_i)$; (2) Cheating in reporting $|h_{i0}|$ or $|h_{0i}|$. In particular, a cheating user i always reports a random value for $|h_{i0}|$ or $|h_{0i}|$; (3) There is a half-and-half mixture of the above two types of cheating users.

In our second experiment, we want to show the influence of cheating behaviors on system throughput in a more general case. Like in the first experiment of system throughput, we test three cheating strategies as well as an honest strat-

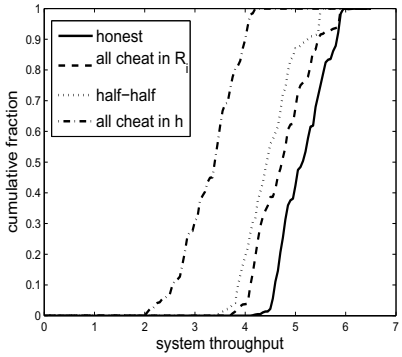

Figure 3.9. CDF of the system throughput in Section 3.2

egy of secondary users. For each strategy, we run the simulation 100 times. And in each run, a secondary user takes random channel parameters.

The cumulative distributions of achievable system throughput are shown in Figure 3.9, and we can see that when all secondary users cheat in reporting h_{0i} and h_{i0}, the average achievable system throughput is the smallest. Only when all users behave truthfully, or in other words have been successfully suppressed by our scheme, can the system throughput be maximized.

The results in Figures 3.8 and 3.9 demonstrate that when the Zhang-Zhang protocol is used, the overall system throughput can be negatively affected by any of these three types of cheating behavior. In contrast, when our scheme is used to completely suppress cheating, the achieved throughput is always close to the above maximum throughput of the Zhang-Zhang protocol, regardless of the number of cheating users. In these experiments, suppressing cheating behavior can improve the total system throughput by up to 69.4% in the face of selfish users.

The results of the experiment above show that suppressing cheating is very useful for improving the total system throughput, when there are selfish users.

Consequently, using a cheat-proof scheme like ours is important.

Fairness Evaluation

In this subsection, we evaluate the starvation percentage and the Jain's fairness index within each spectrum allocation.

First, without loss of generality, we measure the starvation percentage in a setting identical to that of the third set of experiment in utility evaluation, where there are 20 secondary users and one primary user. The starvation percentage of secondary users (i.e., the percentage of secondary users never selected as relay users) is measured in 1000 runs. We allow each secondary user to change their channel gain magnitudes within a range of $\pm 1\%$ (due to movement) after each run so as to simulate a more dynamic scenario. We compare the results of the original version of our extended scheme with that of our starvation-reduced scheme, in which the system parameter $\tau = 80\%$.

Second, under the same simulation settings, we measure the Jain's fairness indices [35] in our system. The fairness index is computed as

$$J(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i^2}. \tag{3.14}$$

In our evaluation, we measure two types of fairness indices: (1) $x_i = R_i t_i$, the allocated transmission rate of user i in each run. (2) $x_i = \sum_j R_{ij} t_{ij}$, user i's accumulative transmission rate in all runs. Specifically, in case (1), $x_i = 0$ when i is not selected as a secondary relay user in one run, and in case (2), $x_i = 0$ if i has never been selected as relay user during all runs. For each type of fairness index, we test 1000 runs. In case (1), we evaluate x_i in each run when our starvation-reduced scheme is implemented and when it is not, and after all

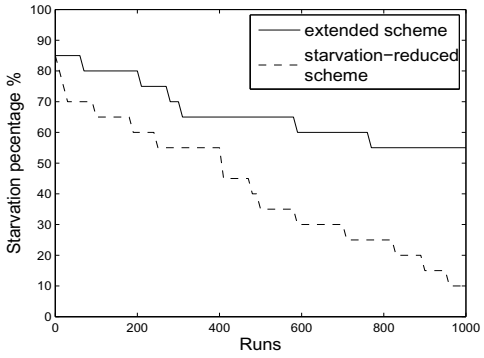

Figure 3.10. Starvation percentage in Section 3.2

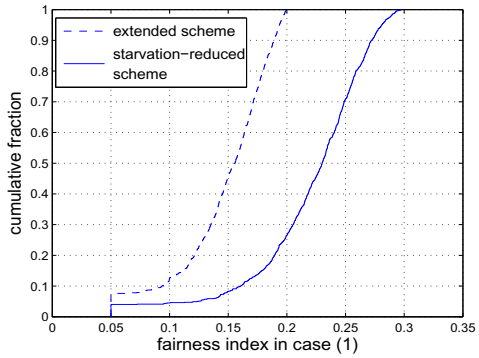

Figure 3.11. The cumulative fraction of Jain's fairness index in case (1) in Section 3.2.

1000 runs we compare the cumulative distribution of x_i for both situations. In case (2), we record i's allocated transmission rate in each run, and compute the fairness index of sum x_i after all 1000 runs.

In Figure 3.10, we can see that with our starvation-reduced scheme, the final starvation percentage after 1000 runs is 10%, which means that 90% of the secondary users (18 users) have participated in the cooperative relay service and have benefited from accessing the primary user's free channel. However, if we merely use the extended scheme, only 45% of secondary users (9 users) have been selected, while 55% users have never been in these 1000 runs.

Figure 3.11 shows that we achieve better Jain's fairness indices when the starvation-reduced scheme is present in case (1). The solid line indicates the results when the starvation-reduced scheme is implemented, while the dotted line indicates the results when the starvation-reduced scheme is absent. The minimum index for both schemes is 0.05 when one and only one secondary user is selected as relay user out of 20 users in one run. The maximum index our starvation-reduced scheme can achieve is 0.3. However, when the starvation-reduced scheme is absent, the maximum index drops to 0.2. The two cumulative fractions at the starting point are different, because it is less likely that only one user is selected as relay user in our starvation-reduced scheme. Although the improved fairness indices are no more than 0.3, far less than 1, we emphasize that our objective in fairness is not to equally share the free spectrum among all secondary users. Instead, we aim to provide opportunities of spectrum access for as many capable secondary users as possible.

In case (2), we measure all secondary users' cumulative transmission rates allocated by the primary user in all 1000 runs. We then compute the Jain's fairness index following Equation (3.14). The result is shown in Table 3.2. In our tests, only 2 users have *never* been selected as relay users when the starvation-reduced scheme is present and up to 12 users fail to become relay candidates when that scheme is absent (as shown in Figure 3.10). Compared to the fairness index under our original extended scheme, the new fairness index has been improved by 83.64%.

Payment

Payment is another important factor in our system. We observe payments of secondary users in this subsection. In this set of experiments, we create a ran-

Table 3.2. The measurement of Jain's fairness index of case (2) in Section 3.2

If the starvation-reduced scheme is present	the value of Jain's fairness index
No	0.3381
Yes	0.6209

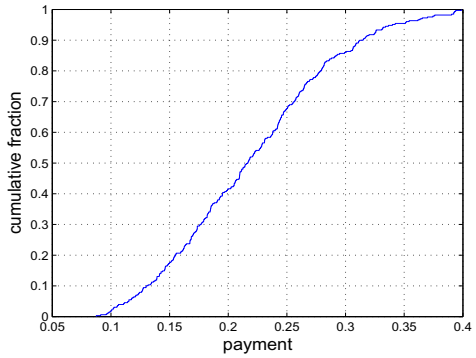

Figure 3.12. The cumulative fraction of all payments in Section 3.2.

dom topology ($2 \leq |SU| \leq 20$) for each test and in each test all secondary users have random channel information under constraints $0 < R_i \leq 10, -80dB < |h_{0i}|, |h_{i0}| \leq -50dB$. System parameter ω is set to be 2 and c is set to be 0.01. We assume all secondary users behave truthfully in each test, because cheating behavior can always be detected under our algorithms and the cheater will never be selected as relay candidate. After the primary user determines the relay candidates, the required payment of each candidate is recorded. We test 1000 runs and record all payments. The cumulative distribution fraction results are shown in Figure 3.12.

In Figure 3.12, among all payments, the smallest value is 0.088 and the largest value is 0.396. We can also see the payments are almost uniformly distributed between 0.088 and 0.396. This result is consistent with our settings in which transmission rate R_i and channel gain h_{0i}, h_{0i} all follow uniform distributions

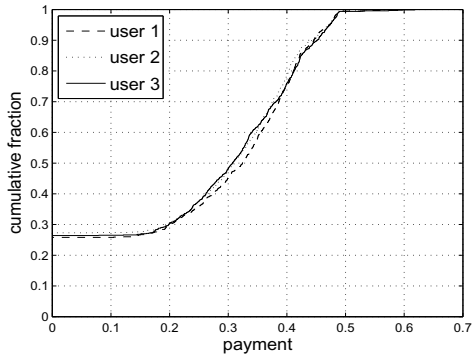

Figure 3.13. The cumulative fraction of three out of five secondary users' payments in Section 3.2.

and are all independent of each other.

Then we record the payments of the first three secondary users when $|SU|$ is set to be 5 (and other settings are kept the same as in above experiments). The cumulative fraction results are shown in Figure 3.13: (1) For any of the three users, the chance of not being selected as relay candidate is about 27%; (2) The minimum payment among all three users is 0.086 which is achieved by user 2; (3) The maximum payment among all three users is 0.612 which is achieved by user 3. The reason why the maximum payments are different in Figure 3.12 and 3.13 is that in Figure 3.12 it is possible that more than 5 secondary users ($k > 5$) are selected as relay candidate which makes the payment less. In summary, all these observations are consistent with our expectations.

3.2.8 Summary

In this work, we study the cheating behavior of selfish users in cooperative relay and present the first cheat-proof scheme to suppress cheating. Theoretically, in the model of strategic game, we rigorously show that with our basic scheme, all

secondary users following the protocol is a DSE. Then we extend our study to the model of the entire cooperative relay process. In the model of an extensive game with perfect information, we show that it is a SPNE for all users to follow our extended scheme. Experimentally, we perform extensive simulations to test the effectiveness of our scheme. The simulation results verify that, without our scheme, selfish users can cheat to harm other users in cooperative relay, but with our scheme, they have incentives to follow the protocol, i.e., not to cheat. Consequently, our schemes improve the system throughput significantly in face of selfish users. In our experiments, it improves the system throughput by up to 69.4%. Furthermore, we consider fairness and present a starvation-reduced scheme.

We stress that, while our work has a lot of theoretical analysis, the problem we study is closely related to some real world scenarios. For instance, consider a scenario in which relay stations are established to support 4G communications between a base station and mobile phones. These relay stations may belong to owners other than the base station operator, and thus may have their own interests that differ from the base station's. So if a relay station can cheat to benefit itself (more precisely, its owner), then it is likely to cheat.

Recall the numerical example in previous section. User 1 in this numerical example could be a selfish relay station as mentioned above. When user 1 cheats as we showed in this example, user 1 can get more access time, though it also needs to make more payment. When the extra access time outweighs the additional payment, user 1 has incentives to cheat—-this is very likely because the relay stations may have been established by an owner who is in need of a large amount of access time.

Although the scope of this work is mainly focused on cognitive radio networks, we notice that with minor modification the proposed algorithms can be applicable to many kinds of cooperative relay networks. But we also stress that certain aspects our scheme (in the current form) may be more suitable for cognitive radio networks than for other networks. One reason is that an identifying feature of cognitive radio networks is the fact that both primary user and secondary user coexist in the same spectrum and share resources without interference or conflict. And the main objective of cognitive radio design is to increase the spectrum utilization among different users. Recall in our system, the primary user and the secondary user share the same spectrum. And the cooperative relay is allowed only when the overall throughput of the primary user is improved. Given other network models, like DTN, the relay mechanism designed in our system cannot be perfectly deployed as in cognitive radio networks. We hope our work will be the first step towards cheat-proof cooperative relay.

3.3 Algorithms for cognitive radio networks using cooperative relay with general utility functions

3.3.1 Background and Motivation

In [81], Simeone et al., present a cooperative relay model that is different from previous auction based models. In their model, secondary users can relay data

for the primary user, and in return the primary user should allocate portions of its free spectrum for those secondary users who provide relay services. Secondary users compete with each other based on their capability of relay transmission rather than currency, and the primary user focuses on maximizing its overall throughput (including throughput on direct link and throughput via secondary users' relay services). This is the first work that introduces Stackelberg game model into cooperative relay problems in CRN, while they also present a nice proof for the existence of Nash Equilibrium (NE).

In [82], Zhang et al., study the cooperative relay in CRN as well but use a different Stackelberg game model. They introduce payments into their model, in which secondary users have to decide their payments to the primary user and their affordable relay transmission rates when competing for the access of primary user's free spectrum. The utilities of primary users and secondary users are affected by their achievable transmission rates as well as the payments. Moreover, in their model, they elegantly prove that following their protocol is a unique NE. In both work, primary user and secondary users are all assumed to be selfish: The primary user limitedly shares its free channel with secondary users, in order to exploit relay service and better utilize the authorized spectrum. On the other hand, each secondary user determines a proper amount of resources (transmission power or transmission rate) used for relay service (and also a proper amount of payment in [82]), such that its utility is maximized once being selected as a relay user. In [83], the authors present cheat-proof schemes based on the same utility models as in [82]. The selfish behaviors thus can be suppressed during cooperative relay. Tang et al., in [61] study the cooperative relay in CRN based on a Stackelberg game model where the utility of the pri-

mary user is defined to be the sum of its weighted satisfaction of data rate and the revenue it obtains from the secondary user, and the utility of the secondary user equals to the weighted satisfaction of data rate minus its payment. In their model, given one primary user and one secondary user, an optimal cooperative spectrum leasing strategy is provided. The entire process of cooperative relay is modeled as a Stackelberg game in [81][82][83][61], while the interactions among secondary users are modeled as a strategic game (which is a part of the Stackelberg game).

One key observation is that although the Stackelberg game is widely used for cooperative relay in CRN, the existing work is all based on one assumption: the utility functions for primary user and secondary users belong to a specific type of functions. For example, in [82] the utility for primary user is a linear function of the achievable transmission rate (and payments). However, in some situations, the primary user's interest in the transmission rate may not be linear. It could be that when the transmission rate is below a threshold, the primary user quickly becomes very unhappy and thus the utility should decrease faster than a linear function. The existing results have not taken such possibilities into consideration and cannot be applied to such situations. Therefore, an efficient model which considers general utility functions in cooperative relay would be very useful in practice. As far as we know, there is no existing work providing solutions for this problem.

In this work, we allow users to have general utility functions in cooperative relay. In other words, there is no special restriction on users' utility functions. In such a scenario, we provide effective algorithms for both primary user and secondary users. Following our algorithms, each secondary user can decide its

optimal strategy when competing with others in cooperative relay, while each primary user can determine a set of secondary user candidates in order to maximize its own utility. To be more precise, we present an iterated algorithm for secondary users, in which they can finally reach a NE in determining their payments to the primary user in a pure strategic game. Also we present a genetic algorithm, namely simulated annealing algorithm, for primary user to search for optimal utility rather than using exhaustive search. Our experiment results show that our algorithms are reliable and efficient (see evaluation section). Our contribution can be summarized as follows:

- We design an iterated algorithm for secondary users to achieve a Nash Equilibrium.
- We design a simulated annealing algorithm for primary user to find its optimal utility.
- We perform simulations. Results demonstrate that, our algorithms are reliable and efficient.

3.3.2 Simulated Annealing

Given a set SU of secondary user candidates, the primary user has to determine a subset S in order to obtain its highest utility. Although the primary user can find S using exhaustive search on possible combinations of secondary users, the time complexity is very high. Therefore we design an algorithm for primary user to find its optimal utility in reasonable time based on simulated annealing.

Simulated annealing is a generic probabilistic metaheuristic for global optimization problems. It is more efficient than exhaustive search algorithm, es-

pecially when the search space is large. Before we present our simulated annealing algorithm for primary users, we have to specify some notations within our model, including the state space, the neighbor of a state and the acceptance probability function.

Denote by a n-bit state $A = a_1 a_2 \dots a_n$ the combination of the selected secondary users, where $n = |SU|$, $a_i = 1$ if secondary user i is selected as a relay node, and $a_i = 0$ otherwise.

Definition 6. *The* state space *SP is the set of all possible states of A except when $A = 00 \dots 0$.*

It is easy to see $|SP| = 2^n - 1$.

Definition 7. *Two states $A_i, A_j \in SP$ are* neighbors *if either of the following is satisfied:*

1. $w(A_i) = w(A_j)$ *and* $H(A_i, A_j) = 2$
2. $|w(A_i) - w(A_j)| = 1$ *and* $H(A_i, A_j) = 1$

where $w(A)$ is the weight[5] *of A and $H(A_i, A_j)$ is the Hamilton distance*[6] *between A_i and A_j. Specifically, define a function $E(A_i, A_j) = 1$ if A_i and A_j are neighbors and $E(A_i, A_j) = 0$ otherwise.*

As we mentioned before, the primary user can compute its utility if the set S of selected secondary users is determined. Therefore, given a state $A_i \in SP$, denote the primary user utility by

$$U_P(A_i) = \Omega((R_{0i})_{i \in S}, (R_{i0})_{i \in S}, (c_i)_{i \in S}),$$

[5] we use weight to represent the number of "1" in a string.

[6] Hamilton distance is the number of different digits between two strings

where S is the set of selected secondary users in state A_i.

Definition 8. *The probability of making the transition from the current state A_i to a candidate new state A_j is specified by an acceptance probability function $P(U_P(A_i), U_P(A_j), T)$, where T is a global time-varying parameter T. The* acceptance probability *function can be calculated as follows:*

$$P(U_P(A_i), U_P(A_j), T)$$
$$= \begin{cases} 1 & U_P(A_j) > U_P(A_i) \\ e^{\frac{U_P(A_j) - U_P(A_i)}{T}} & U_P(A_j) \leq U_P(A_i). \end{cases}$$

3.3.3 Nash Equilibrium Search Algorithm for Secondary Users

Given a set of secondary user competitors, each secondary user determines its payment to the primary user in order to maximize its own utility. In this work we consider a pure strategy game, in which Nash Equilibrium exists when the following conditions are satisfied [43]:

- Each C_i is a compact and convex subset of Euclidean space.
- Each u_i is quasiconcave in c_i and continuous.

Define $c_{-i} = (c_1, c_2, \ldots, c_{i-1}, c_{i+1}, c_m)$, $(i = 1, 2, \ldots, m)$. To find the NE, it suffices to find the strategy set $c^* = (c_1^*, c_2^*, \ldots, c_m^*)$ such that for any secondary user i, $u_i^* = \pi(R_i, (c_i^*, c_{-i}^*)) \geq u_i = \pi(R_i, (c_i', c_{-i}^*))$, where $c_i^* \neq c_i'$. In another word, if secondary user i has knowledge of other users' strategies, then it can compute its payment as $c_i^* = b(c_{-i}^*)$ to maximize its utility. If there is at least one NE in

the game, all secondary users can *repeatedly* perform their best response functions and share the results time after time before each secondary user i reaches its best strategy c_i^*:

$$c_i^{k+1} = b_i(c_{-i}^k), \tag{3.15}$$

here k is the sequence number of iterated performance of all secondary users.

Each secondary user can obtain its best response function from its utility function. One possible way to find $b_i()$ is under the situation that the partial derivative of the utility function $u_i = \Theta(R_i, (c_j)_{j \in S})$ with respect to c_i is zero. In another word, if other secondary users do not change their payments in next round of iteration, $c_i = b_i(c_{-i})$ is the payment where all users reach a Nash Equilibrium.

We assume each secondary user can exchange information with others using a reliable channel. Before each secondary user runs our algorithm, he should know other users in S. Also he should be able to compute $b_i()$ given a specified utility function $\Theta()$. A synchronized timer T_{max} is also required for the system to restart the algorithm if it does not converge in a certain amount of time. Based on these assumptions, our algorithm works as follows: In the first round, each secondary user initiates a random non-zero payment and broadcast it to others. Whenever he receives other $|S| - 1$ payments, he should compute its best payment for the next round using its best response function and broadcast the result. He should follow these steps repeatedly until the payments made by other users as well as himself do not change, which means the payments of all secondary users have been decided. In this way, a convergence point is found where all secondary users reach a NE at the same time. If after T_{max} rounds, the

secondary users can not determine their payments, the algorithm will restart and each secondary user initializes a new non-zero payment. The details of the algorithm can be found in Algorithm 7.

Algorithm 7 Payment decision algorithm of user i

Input: the set S of secondary users, user i's transmission rate R_i and its best response function $b_i()$, an initial set of non-zero payments $c^0 = (c_1^0, c_2^0, \ldots, c_n^0), n = |S|$, an error threshold ϵ and a time threshold T_{max}.
Secondary user i does:

$c^* = c^0$, $u_i^* = \Theta(R_i, c^0)$
$T = 1$
while $T \leq T_{max}$
    if $T < T_{max}$
        $c_i' = b_i(c_{-i}^*)$
    else
        $T = 1$
        user i generate a new non-zero c_i'
    broadcast c_i' to other secondary users
    wait until receive all $c_j'(j \in S, j \neq i)$
    $c' = c_i' \cup c_{-i}'$
    if for every $|c_j' - c_j^*| < \epsilon$, $j \in S$ then
        break
    else
        $c^* = c'$, $T = T + 1$
end while
Output: c^*, u_i^*

3.3.4 Heuristic Algorithm for Primary Users

Although the primary user can simply run an exhaustive search for its optimal utility when a new game is modeled, the solution can be very time consuming. Our experiment result shows that, even for a simple utility function, when $|S| =$

30, the decision process will take up to 15 seconds (see Figure 3.16), which is clearly unrealistic. Therefore, an efficient algorithm is needed for the primary user to search for its optimal utility in practice.

In this section we present an algorithm based on simulated annealing. In previous section, we reviewed the basic concepts and notations from stimulated annealing theory. Now suppose $G(SP, E)$ is an undirected graph whose vertices are all states in SP, and in this graph there is an edge between A_i and A_j if $E(A_i, A_j) = 1$.

Lemma 2. *The diameter of* $G(SP, E)$ *is* $(n-1)$, $n = |SU|$.

Proof. We randomly pick two states A_i and A_j, consequently we have two selected secondary sets S_i and S_j. Define $t_1 = S_i - S_j$ and $t_2 = S_j - S_i$. Thus the shortest distance between these two states in G is $D_{ij} = \max\{|t_1|, |t_2|\} \leq n-1$. Thus, the diameter of $G(SP, E)$ is $\max_{A_i, A_j \in SP}\{D_{ij}\} = n-1$. □

Based on Lemma 2, we know the shortest distance between two arbitrary states is small, which indicates simulated annealing is applicable to our design of state space for this problem.

Our algorithm for primary user works as follows: the primary user initiates a random state A_0 at the very beginning. After that, all secondary users selected as relay nodes in state A_0 are notified. These secondary users have to perform Algorithm 7 so as to decide their payments, and report the results together with their relay transmission rates to the primary user. The primary user can then compute its utility based on the knowledge of these information. At each following step, the primary user considers a neighboring state A' of the current state A, and probabilistically decides between moving to state A' or staying still

in state A. These probabilities ultimately lead the system to move to states of more utility. The steps are repeated until a given countdown timer T_{max} has been exhausted.

The probability of making a transition from the current state A to a candidate new state A' is determined by the acceptance probability function $P(U(A), U(A'), T_{max})$, which is specified in previous section. Note that the output of the function is nonzero when $U(A') < U(A)$, meaning that the system may move to the new state even when the utility is smaller than the current one. It prevents the algorithm from becoming stuck in a local maximum utility rather than the global optimal utility. Moreover, when T_{max} goes to zero, the probability $P(U(A), U(A'), T)$ must tend to zero if $U(A') < U(A)$. Therefore, in the early stage of the algorithm, the system is willing to take moves that goes to smaller utility, however prefers to staying in current state with larger utility when time goes near the end. The details of this algorithm is given in Algorithm 8.

Note in Algorithm 8, whenever the primary user reaches a new state (a set of secondary users), those selected secondary user should follow Algorithm 7 to decide and report their payments.

3.3.5 Evaluations

In this section, we provide evaluations of our algorithms in various scenarios. Throughout the experiments, we want to show that both of our algorithms are efficient in practice, and also we try to demonstrate the accuracy of our algorithm in searching for primary user's optimal utility.

To achieve these goals, we design our experiments as follows: (1) In the first set of experiments, we randomly choose a subset of SU. Each secondary

Algorithm 8 Heuristic search for primary user's optimal utility

Input: the set of secondary user SU, an initial state A_0, a countdown timer T_{max}, a function $A' = neighbor(A)$ generating a random neighbor A' given the state A, and a function $rand()$ producing a random number within (0,1).
Primary user does:

$A^* = A = A_0$, $u^* = u = U_P(A_0)$

$T = T_{max}$

while $T > 1$

$A' = neighbor(A)$, $u' = U_P(A')$

if $rand() < P(u, u', T)$ then

$A = A'$, $u = u'$

if $u^* < u$ then

$A^* = A$, $u^* = u$

$T = T - 1$

end while

Output: A^*, u^*

user selected should follow Algorithm 7 to decide its payment and finally all users reach a NE. We want to show that the algorithm is efficient. (2) In the second set of experiments, the primary user should run Algorithm 8 to search for its optimal utility. And we study the accuracy of the output as well as the overhead of the algorithm. To better illustrate the efficiency and effectiveness of our algorithms, we compare the overhead and accuracy of our algorithms with those of exhaustive search.

In this work we consider general utility function of primary user and secondary users, hence in this part we should specify proper utility functions for evaluation. Recall the condition under which a Nash Equilibrium exists in a pure strategic game in previous section. In this evaluation, without loss of gen-

erality we use one possible utility function of secondary users:

$$u_i = w\sqrt{R_i(\frac{\sum_{j\in S, j\neq i} c_j}{|S|} + c_i)} - c_i, \tag{3.16}$$

Given the utility function above, the best response function of secondary user i can be obtained:

$$b_i(c_{-i}) = \frac{w^2 R_i}{4} - \frac{\sum_{j\in S, j\neq i} c_j}{|S|} \tag{3.17}$$

Moreover, we define the utility function of primary users as:

$$U_P(S) = wR_P(S) + \sum_{i\in S} c_i, \tag{3.18}$$

where

$$R_P(S) = \min\{R_{PS}(S), R_{SP}(S)\}, \tag{3.19}$$

$$R_{PS}(S) = \log_2(1 + \min_{i\in S} R_{i0}), \tag{3.20}$$

$$R_{SP}(S) = \log_2(1 + R_0 + \sum_{i\in S} R_{0i}), \tag{3.21}$$

Overhead

To demonstrate the efficiency of our algorithms, we use the above two sets of experiments to evaluate the overhead of Algorithm 7 and Algorithm 8. Here the evaluated overhead is defined as the overall time the targeted algorithm uses to achieve an output given valid inputs.

In our first set of experiment, we generate a set SU of secondary users with random datasets including the transmission rates $0 < R_i, R_{0i}, R_{i0} < 10$ and the

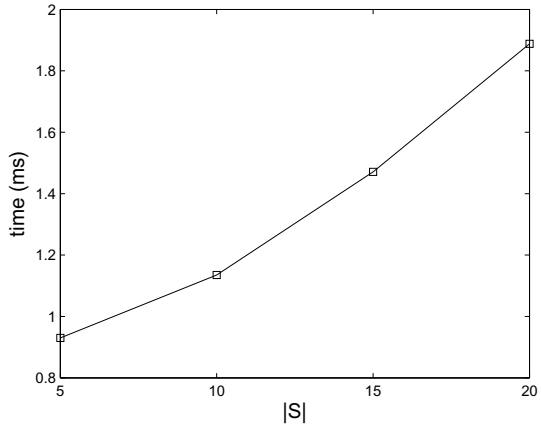

Figure 3.14. Average overhead of algorithm 7 in Section 3.3.

initial payment $0 < c_i < 2$ for each secondary user i. $|SU| = 50$. Then we choose a random subset S out of SU and ask each selected user in S to run Algorithm 7.

We consider the efficiency when $|S| = 5, 10, \ldots, 20$. We randomly pick a secondary user to record the overhead of Algorithm 7 (the overhead must be the same for all selected users, and in each experiment we pick the user with the least index number). The error threshold parameter ϵ is set to be 0.001.

The result is shown is Figure 3.14. Note we do not show average overhead of exhaustive search algorithm here. The reason is that the overhead of exhaustive search for NE is not tolerable, which takes up to several seconds when $|S| = 20$. Theoretically, the overhead of exhaustive search for NE grows exponentially when $|S|$ increases. On the other hand, the average overhead of our Algorithm 7 is no more than 2ms even if $|S| = 20$, which is a large number of secondary users in practice. Therefore, we know Algorithm 7 is much more efficient than exhaustive search.

In our second set of experiments, to better illustrate the advantage of using simulated annealing algorithm, we compare the overhead of Algorithm 8 with a

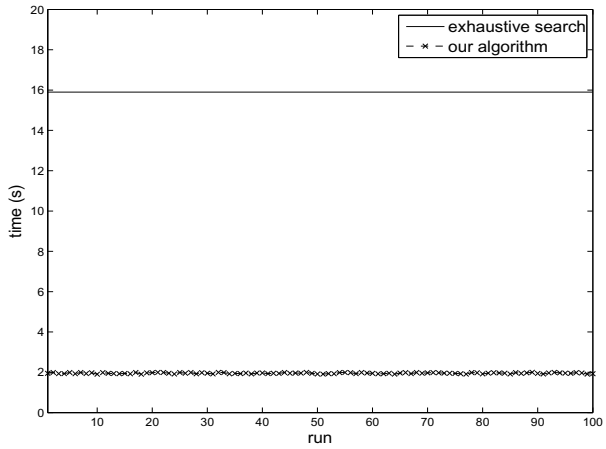

Figure 3.15. Overhead of algorithm 8 when $|SU| = 30$ in Section 3.3.

modified exhaustive search algorithm which also uses Algorithm 7 to determine each secondary user's payment but exhaustively search the state space SP. Otherwise, the overhead will become so large that the comparison is meaningless. The details are as follows: We generate random datasets for secondary users in each experiment as in the first set of experiment. First, the overall overhead of Algorithm 8 is recorded in 100 runs when $|SU| = 30$. And we compare the result with that in the modified exhaustive search algorithm. Second, we consider a more general case when $|SU| = 5, 10, \ldots, 50$, and measure the average overhead of each case for primary user to reach its optimal utility. We also compare the overhead of our algorithm with that in the modified exhaustive search algorithm.

In Figure 3.15, the result shows that our algorithm only takes 2 seconds to output while the exhaustive search algorithm will take almost 16 seconds when $|SU| = 30$. The overhead is reduced by 88% in this situation. The more secondary user compete for relay candidates, the more efficient our algorithm will be.

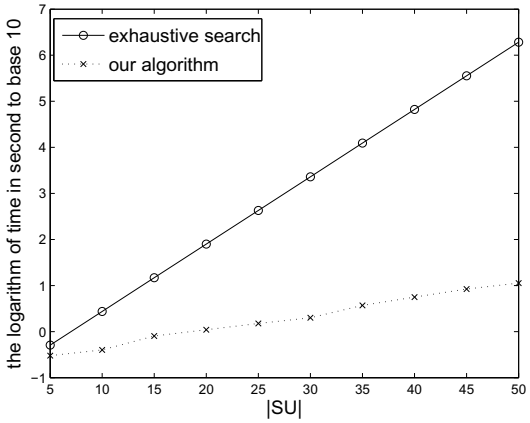

Figure 3.16. Comparison between algorithm 8 and exhaustive search in Section 3.3.

Moreover, we compare the average time used in our algorithm with the time in exhaustive search in Figure 3.16. The result shows that the more available secondary users in SU, the more efficient our algorithm is. Note we use the logarithm of time to base 10 instead of time in second in this figure. When $|SU| = 50$ the overhead of exhaustive search is more than 10^6 seconds, while it only takes $10^{0.3} = 2$ seconds to find the optimal utility using Algorithm 8. Because in our model, the primary user does not need to traverse the entire search space to find the optimal solution, the overhead of Algorithm 8 is reasonable.

User Utility

In this part of evaluation, we want to show the accuracy of our algorithms. For Algorithm 7, we use the same set of experiment when evaluating the overhead of Algorithm 7 and the time threshold T_{max} is set to be 500. We obtain the optimal strategies of users through exhaustive search and compare the output of our algorithms with those optimal values. The results show that our algorithm can always find the NE before timer reaches T_{max}. Therefore, the accuracy

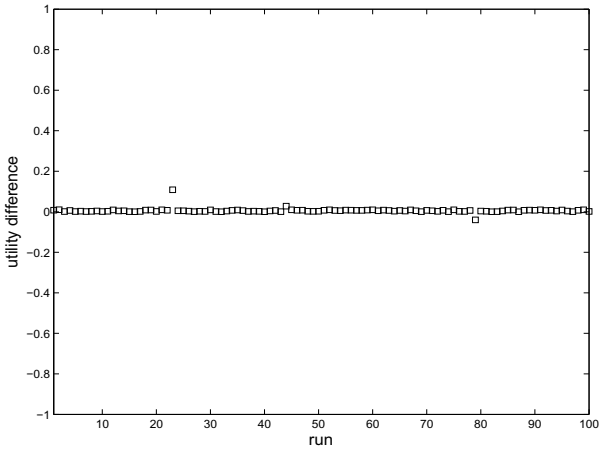

Figure 3.17. Utility difference when $T_{max} = 500$ and $|SU| = 30$ in Algorithm 8 in Section 3.3.

of Algorithm 7 is 100%.

To show the accuracy of Algorithm 8, we consider a set of secondary users $|SU| = 30$ and also we compute the real optimal utility u_p of primary user. Then we let primary user run Algorithm 8 100 times when the time threshold T_{max} is set to be 500. We compare the output of each run with u_p to show how our algorithm performs within limited time threshold.

In Figure 3.17, we show the utility difference between real optimal utility of primary user and the output of Algorithm 8 when the time threshold $T_{max} = 500$ and $|SU| = 30$. The result shows that even when the time threshold is set to be a small number, most outputs (97 out of 100) stay in the acceptable range of $\pm 1\%$ of optimal utility.

3.3.6 Summary

In all existing game theory based cooperative relay models in cognitive radio networks, the utility functions for primary users and secondary users are in

specified types. Hence, one model may not be applicable for different scenarios. In this work, we consider the general case of utility functions of primary user and secondary users. First we present an algorithm for secondary users to reach a NE in determining their payments to the primary user. Following this algorithms, secondary users can decide their optimal strategy when competing with others in our model. Then we present another algorithm for primary user to search for the optimal utility instead of using exhaustive search. The experiment results show that our algorithms are accurate and efficient. The overhead by our algorithms can be reduced by up to 88% compared with exhaustive search.

3.4 MICOR: A Market for Incentive-Compatible Cooperative Relay in Cognitive Radio Networks

3.4.1 Background and Motivation

Here we study the incentives of secondary users in the second type of cooperative relay (see Section Technical Preliminaries), which is among the secondary users only and does not involve any primary user.

A crucial problem in this cooperative relay process lies in the (lack of) incentives for the potential relay nodes, i.e., the secondary users who have spare spectrum. Why are these nodes willing to relay traffic of other secondary users using their own communications resources? In general, such relay does not

benefit the relay nodes but can consume significant amounts of their resources. These nodes should have little or no interest in providing relay services, unless they receive appropriate compensation for such services.

In this work, we propose that relay nodes should receive payments for their relay services, so that they have incentives to provide such services. Note that payment is a standard approach for stimulating cooperation. In wireless networks, the payment approach has been used to stimulate cooperation in many scenarios such as routing and packet forwarding of ad hoc networks [72]-[76], channel assignment [77]-[80], and most importantly, other types of cooperative relay [82]-[89].

Specifically, we design MICOR, a Market for Incentive-compatible COoperative Relay. MICOR allows each potential relay node to submit a valuation function, which describes how much payment its relay service is worth. (This information must be represented by a function rather than by a single number, because for each possible amount of relay service, a corresponding amount of payment needs to be specified.) Then, based on the submitted valuation functions, the session that needs cooperative relay selects the relay nodes to be used, computes the amount of relay traffic for each selected relay node, and also decides the corresponding amount of payment for each selected relay node.

One advantage of MICOR is that it guarantees all potential relay nodes have very strong incentives to provide their true valuation functions. In other words, MICOR prevents these node from lying about their valuation functions. This is extremely important because as we mentioned above, the selection of relay nodes, the assignment of relay traffic, and the computed payment all depend on the valuation functions. More precisely, MICOR guarantees that it is a Dom-

inant Strategy Equilibrium (DSE) for all potential relay nodes to submit their true valuation functions.

The above guarantee has one precondition (called *No Monopoly* condition hereafter; see analysis section for definition): There is no indispensable relay node for the cooperative relay. That is, with any single relay node excluded, the cooperative relay can still succeed. Whether this condition holds depends on the assignment of channel (and the demand for relay traffic). Consequently, we also design an algorithm for channel assignment that reduces monopoly.

Our contributions can be summarized as follows.

- We design MICOR for cooperative relay among secondary users. We rigorously prove that, if MICOR is used, it is a DSE for all potential relay nodes to report their true valuation functions, assuming there is no monopoly.
- For MICOR, we also design a channel assignment algorithm that reduces monopoly.
- Results of extensive evaluations demonstrate that MICOR provides strong incentives to potential relay nodes, effectively reduces monopoly, and has good efficiency.

3.4.2 System Model

Suppose there is a cognitive radio network with primary users and secondary users, where secondary users can access primary users' spectrum holes opportunistically. Among the secondary users, there are a source node SN and a destination node DN. We consider the session from SN to DN, which may need

other nodes to provide cooperative relay for better performance. Also among the secondary users, there are N (potential) relay nodes for the session between SN and DN. Let the relay node set be $RNS = \{RN_i | i = 1, \ldots, N\}$, where RN_i represents the ith relay node. All these relay nodes are within the communication ranges of both SN and DN, so that if they are willing to, they can relay data from SN to DN.

Now consider channel assignment among these nodes. Just as in [67], we assume that each involved node has a cognitive radio that can dynamically access any combination of available channels. Also suppose that there are K channels in total ($K > 1$), where all channels have equal bandwidths. Assume that the channels for direct transmission from SN to DN have been assigned, and so we can focus on the channel assignment for relay only. Denote the channel assignment for cooperative relay by $Z = \{z_{i,j,k}\}_{1 \le i,j \le N+2, 1 \le k \le K}$, where $z_{i,j,k} = 1$ if channel k is assigned to the communication link from RN_i to RN_j, and 0 otherwise. Here, two special indices above N are reserved for the source and destination: We define $RN_{N+1} = SN$ and $RN_{N+2} = DN$, for simplicity of presentation.

The channel assignment Z needs to satisfy the following requirements:

- A channel can be assigned to a communication link only if this channel is available to both endpoints of the link. (Note that a channel could become unavailable at a certain location because the channel is used by a primary user, because the channel is used by the direct transmission, or because there is interference from other sessions). Formally, let $a_{i,k} = 1$ if channel k is available to RN_i, and $a_{i,k} = 0$ otherwise. Then, for all i, j ($1 \le i, j \le$

$N+2$), and all k ($1 \leq k \leq K$),

$$a_{i,k}a_{j,k} = 0 \Rightarrow z_{i,j,k} = 0.$$

- The assignment should not cause interference among the involved nodes (SN, DN, and the nodes in RNS). Let us adopt an assumption from [67]: All these involved nodes are in the interference range of each other. Then, each channel is assigned to at most one communication link: For all k ($1 \leq k \leq K$),

$$\sum_{1 \leq i,j \leq N+2} z_{i,j,k} \leq 1.$$

Suppose that the session between SN and DN has demand of rate R_D. Let C_D be the capacity of direct transmission from SN to DN. If R_D is greater than C_D, then the session definitely needs cooperative relay in order to satisfy the demand. In this case, $R_R = R_D - C_D$ is the amount of additional transmission rate needed to meet the session's demand besides direct transmission.

To reward those relay nodes who provide relay services, the session should provide payments to them as compensation. Each relay node RN_i has a valuation function $V_i()$ of the relay service it may provide, which describes the amount of payment its relay service is worth. For any relay transmission rate f, RN_i believes that it deserves a payment of $V_i(f)$ for providing the relay service. This valuation function $V_i()$ depends on a number of factors, such as the channel assignment and node RN_i's own demand of data transmission. There are a few requirements that a reasonable valuation function $V_i()$ should satisfy. Specifically, $V_i()$ should be a function defined on the interval $[0, c_i)$,[7] where c_i is

[7]Note that c_i depends on the channel assignment, and other factors like node RN_i's own

the maximum transmission rate from SN to DN through the relay of RN_i (*not* including the transmission rate on the direct link from SN to DN), such that:

- (RQ1) $V_i()$ should be increasing, because when the relay node RN_i provides more relay service, RN_i deserves more payment as reward.
- (RQ2) $V_i()$ should be convex, because the utility of secondary users is generally assumed to be concave on f. In other words, when RN_i has used more of its own communications resources to provide relay, RN_i becomes less willing to spend additional communications resources on the relay.
- (RQ3) $V_i()$ should be continuously differentiable.

Besides these requirements, we assume that the relay nodes are not willing to spend all their communications resources on relay (because, e.g., they may have their own demand of data transmission). Formally, we have (RQ4): $\lim_{f \to c_i} V_i(f) = +\infty$, and $\lim_{f \to c_i} V_i'(f) = +\infty$.

Hereafter, for ease of presentation and analysis, we extend the domains and ranges of $V_i()$ and $V_i'()$ by defining

$$\forall f \geq c_i, V_i(f) = V_i'(f) = +\infty.$$

Note that for any $0 \leq f < c_i$, $V_i(f)$ and $V_i'(f)$ are still finite real numbers. Since this extension should be done to all valuation functions, we call it (RQ5).

Sometimes, we use V to represent the profile $\{V_i\}_{i=1}^N$ of valuation functions. Hence, RN_i's utility is

$$U_i = P_i - V_i(F_i), \tag{3.22}$$

demand of data transmission.

where F_i is the relay rate RN_i needs to provide to the session, and P_i is the amount of payment RN_i receives as reward. Our main objective in this work is to design an algorithm that computes F_i and P_i for each relay node RN_i (and also an appropriate channel assignment algorithm that supports the former algorithm).

Clearly, the total data rate from all relays should be R_R, i.e.,

$$\sum_{i=1}^{N} F_i = R_R.$$

If an algorithm computing all the F_i and P_i always satisfies this condition, then we say the algorithm is *feasible*. We stress that feasibility is the basic requirement for any algorithm computing all the F_i and P_i.

As mentioned above, we need to design an algorithm for the session to determine all the F_i and P_i. In addition to being feasible, this algorithm also needs to satisfy the following condition: In order to compute all the F_i and P_i, the source needs the relay nodes to provide information about their valuation functions. However, since the relay nodes are selfish, they may provide false information about their valuation functions. Denote by $W_i()$ the valuation function reported by RN_i to the source. In general, $W_i()$ may not be equal to $V_i()$. But we target to give each relay node incentives to report $W_i() = V_i()$.

3.4.3 MICOR Algorithm Design

In this section, we design the main algorithm for MICOR, which computes all F_i and P_i from the reported valuation functions, assuming that the channels have already been assigned appropriately. We study the assignment of channels in a

later section.

1. if $\exists W_i$ does not meet any of the requirements
   (RQ1)-(RQ5), report cheating; terminate.
2. for $i = 1$ to N
3. $\quad \bar{W}'_i = (W'_i)^{-1}$;
4. Solve the equation
5. $\quad \sum_{i=1}^{N} \bar{W}'_i(x) = R_R$;
6. if no solution, output "No Solution" and terminate;
   otherwise, let the solution be x^*;
7. for $i = 1$ to N
8. $\quad F_i \leftarrow \bar{W}'_i(x^*)$;
9. for $i = 1$ to N
10. $\quad$ Solve the equation
11. $\quad\quad \sum_{j\neq i}^{1\leq j\leq N} \bar{W}'_j(x) = R_R$;
12. $\quad$ if no solution, output "No Solution" and
    terminate; otherwise, let the solution
    be x_i^*;
13. $\quad P_i \leftarrow \sum_{j\neq i}^{1\leq j\leq N} (W_j(\bar{W}'_j(x_i^*)) - W_j(F_j))$.

Figure 3.18. MICOR main algorithm: computing all F_i and P_i in Section 3.4

As illustrated in Figure 3.18, the algorithm works as follows. First, SN collects valuation functions from all relay nodes and checks whether these functions satisfy the five requirements (RQ1)-(RQ5) in the system model. If any of them does not satisfy any of these requirements, the corresponding relay node must have been cheating.

Then, for each valuation function W_i, SN computes the inverse function $\bar{W}'_i$ of W'_i, where W'_i is the derivative of W_i. (Note that $\bar{W}'_i$ is *not* the derivative of $\bar{W}_i$.) Given all $\bar{W}'_i$, SN should find a solution of the equation in line 5 of Figure 3.18. If no solution can be found, then it means that all relay nodes' total achievable transmission rates are still below the amount of additional transmission rate needed by the session. In other words, the capacity of all relay channels is insufficient for the source's relay traffic demand. In this case, the source simply

reports that there is no way to satisfy the demand of this session.

If a solution is found for the equation in line 5, this solution is actually a universal marginal price for relay service. Each relay node's amount of relay traffic can be decided by this marginal price. Finally, to obtain the payment P_i for each relay node RN_i, SN has to first find a solution of equation in line 11. It might be the case that there is no solution to this equation. In this case, a *monopoly* (See Definition 9) is detected. As we show in the next section, our algorithm works under the assumption of no monopoly. Hence, in this case, the algorithm again reports that there is no way to satisfy the demand.

3.4.4 MICOR Algorithm Analysis

Given the MICOR main algorithm designed in the previous section, we now formally show that it is feasible (Proposition 10) and provides strong incentives for relay nodes to report true valuation functions (Theorem 4).

Hereafter, denote by $F_j(W_i, W_{-i})$ the data rate of RN_j's relay when RN_i uses strategy W_i and all relay nodes except RN_i use W_{-i}. Similarly, we use $P_j(W_i, W_{-i})$, $x^*(W_i, W_{-i})$, and $x_j^*(W_i, W_{-i})$ to represent the values of P_j, x^*, and x_j^* when RN_i uses strategy W_i (resp., V_i) and other relay nodes use W_{-i}.

Using these notations, we can now formally define monopoly and no monopoly. Intuitively, a monopoly of a relay node means this node can not be excluded in order for the cooperative relay to be successful. No monopoly means there is no such node and thus cooperative relay can be successful with any single relay node excluded.

Definition 9. *We say there is a monopoly of relay node RN_i ($1 \leq i \leq N$) if*

$$\sum_{1 \leq j \leq N, j \neq i} c_j \leq R_R.$$

We say there is no monopoly if no relay node RN_i ($1 \leq i \leq N$) satisfies the above requirement.

As we mentioned earlier, no monopoly is a precondition for our MICOR main algorithm to be feasible.

Proposition 10. *If there is no monopoly, then the MICOR main algorithm is feasible, i.e., for any W, it always holds that*

$$\sum_{i=1}^{N} F_i(W) = R_R.$$

Proof. First, let us show that no monopoly implies equations in line 11 have solutions. For any i ($1 \leq i \leq N$), because of no monopoly of RN_i,

$$\sum_{1 \leq j \leq N, j \neq i} c_j > R_R.$$

Now let

$$\alpha = \frac{R_R}{\sum_{1 \leq j \leq N, j \neq i} c_j},$$

so we know that $\alpha < 1$, and that

$$\sum_{1 \leq j \leq N, j \neq i} \alpha c_j = R_R.$$

Consequently, $\{x_j^{(0)}\}_{1\leq j\leq N, j\neq i}$ where $x_j^{(0)} = W_j{}'(\alpha c_j)$ is a solution to the equation

$$\sum_{j\neq i}^{1\leq j\leq N} \bar{W}_j'(x_j) = R_R. \tag{3.23}$$

Since $0 \leq \alpha c_j < c_j$, each $x_j^{(0)}$ is a finite real number. Next, we construct a sequence $\{x_j^{(1)}\}_{1\leq j\leq N, j\neq i}$, $\{x_j^{(2)}\}_{1\leq j\leq N, j\neq i}$, ...as follows. Let $J = \arg\max_j x_j^{(\theta)}$ and $J' = \arg\min_j x_j^{(\theta)}$. If $x_J = x_{J'}$, then for all j, $x_j^{(\theta+1)} = x_j^{(\theta)}$. Otherwise, define a function

$$Y(x) = W_{J'}'(\bar{W}_J'(x_J^{(\theta)}) + \bar{W}_{J'}'(x_{J'}^{(\theta)}) - \bar{W}_J'(x)).$$

Clearly,

$$Y(x_J^{(\theta)}) = x_{J'}^{(\theta)} < x_J^{(\theta)}. \tag{3.24}$$

On the other hand, we can pick an arbitrary finite real number $y^{\triangle} > x_J^{(\theta)}$, and let

$$x^{\triangle} = W_J'(\bar{W}_J'(x_J^{(\theta)}) + \bar{W}_{J'}'(x_{J'}^{(\theta)}) - \bar{W}_{J'}'(y^{\triangle})).$$

Clearly, $Y(x^{\triangle}) = y^{\triangle} > x_J^{(\theta)} > x_{J'}^{(\theta)}$. It is easy to see $Y()$ is a decreasing function, and thus $x^{\triangle} < x_J^{(\theta)}$. Hence,

$$Y(x^{\triangle}) > x^{\triangle}. \tag{3.25}$$

Combining Equations (3.24) and (3.25), and taking into account that $Y()$ is continuous, we get that there is $x^{\diamond}$ ($x^{\triangle} < x^{\diamond} < x_J^{(\theta)}$) such that $Y(x^{\diamond}) = x^{\diamond}$. So we let $x_J^{(\theta+1)} = x_{J'}^{(\theta+1)} = x^{\diamond}$. For any other j ($j \neq i, J, J'$), we let $x_j^{(\theta+1)} = x_j^{(\theta)}$. It is easy to verify that $\{x_j^{(\theta+1)}\}_{1\leq j\leq N, j\neq i}$ is still a solution to Equation (3.23), as long as $\{x_j^{(\theta)}\}_{1\leq j\leq N, j\neq i}$ is a solution to Equation (3.23).

It is not hard to show that for any $j_1 \neq j_2$ ($j_1, j_2 \neq i$), $\lim_{\theta\to\infty} x_{j_1}^{(\theta)} = \lim_{\theta\to\infty} x_{j_2}^{(\theta)}$. ∎

Hence, for any $j \neq i$ ($1 \leq j \leq N$), $\lim_{\theta \to \infty} x_j^{(\theta)}$ is a solution to the equation in line 11.

Second, using a method similar to the above, we can show that no monopoly also implies the equation in line 5 has solutions.

Finally, now we know that if there is no monopoly, then all equations in MICOR main algorithm have solutions. From line 8 of the algorithm we know that

$$\forall i, F_i(W) = \bar{W}_i'(x^*(W)).$$

Since $x^*(W)$ is the solution to the equation in line 5, we can plug this into the equation, and so get that

$$\sum_{i=1}^{N} F_i(W) = R_R.$$

Hence, the MICOR main algorithm is feasible. □

So far we have proved the feasibility of our main algorithm. In order to prove that it provides incentives for reporting true valuation functions, we need to first establish a lemma.

Lemma 3. *Suppose that MICOR main algorithm is executed in a network with no monopoly. If $F_i(W_i, W_{-i}) < F_i(V_i, W_{-i})$, then for any $j \neq i$, $F_j(W_i, W_{-i}) > F_j(V_i, W_{-i})$. If $F_i(W_i, W_{-i}) > F_i(V_i, W_{-i})$, then for any $j \neq i$, $F_j(W_i, W_{-i}) < F_j(V_i, W_{-i})$. If $F_i(W_i, W_{-i}) = F_i(V_i, W_{-i})$, then for any $j \neq i$, $F_j(W_i, W_{-i}) = F_j(V_i, W_{-i})$.*

Proof. The proof for the first part goes as follows: If $F_i(W_i, W_{-i}) < F_i(V_i, W_{-i})$, then by Proposition 10, we know that

$$\sum_{j \neq i}^{1 \leq j \leq N} F_j(W_i, W_{-i}) > \sum_{j \neq i}^{1 \leq j \leq N} F_j(V_i, W_{-i}).$$

Hence, there exists $j^* \neq i$ such that $F_{j^*}(W_i, W_{-i}) > F_{j^*}(V_i, W_{-i})$. Combining this with lines 7 and 8 of MICOR main algorithm we get that

$$\bar{W}'_{j^*}(x^*(W_i, W_{-i})) > \bar{W}'_{j^*}(x^*(V_i, W_{-i})).$$

Since $W_{j^*}()$ is convex,

$$x^*(W_i, W_{-i}) > x^*(V_i, W_{-i}).$$

For all $j \neq i$, since $W_j()$ is convex,

$$\bar{W}'_j(x^*(W_i, W_{-i})) > \bar{W}'_j(x^*(V_i, W_{-i})),$$

which is equivalent to that $F_j(W_i, W_{-i}) > F_j(V_i, W_{-i})$.

The proof of the second part is similar to the first part and thus we skip it.

The proof of the third part goes as follows: Assume, for the purpose of contradiction, that $F_i(W_i, W_{-i}) = F_i(V_i, W_{-i})$, but $\exists j$ s.t. $F_j(W_i, W_{-i}) \neq F_j(V_i, W_{-i})$. Then there are two cases.

Case A: $F_j(W_i, W_{-i}) < F_j(V_i, W_{-i})$. By Proposition 10, we know that $\exists k$ such that

$$F_k(W_i, W_{-i}) > F_k(V_i, W_{-i}), \tag{3.26}$$

because otherwise the total demand of R_R can not be satisfied. From lines 7 and 8 of MICOR main algorithm, we get that

$$\bar{W}'_j(x^*(W_i, W_{-i})) < \bar{W}'_j(x^*(V_i, W_{-i})).$$

Since $W_j()$ is convex,

$$x^*(W_i, W_{-i}) < x^*(V_i, W_{-i}).$$

Therefore,

$$\bar{W}'_k(x^*(W_i, W_{-i})) < \bar{W}'_k(x^*(V_i, W_{-i})),$$

which is equivalent to that $F_k(W_i, W_{-i}) < F_k(V_i, W_{-i})$. But this is contradictory to Equation (3.26).

Case B: $F_j(W_i, W_{-i}) > F_j(V_i, W_{-i})$. Similar to Case A, we can show this also leads to a contradiction.

In summary, there is always a contradiction. Hence, the third part is also proved. □

Using Lemma 3, now we can show:

Theorem 4. *Suppose MICOR main algorithm is executed in a network with no monopoly. Then it is a DSE for all relay nodes to report their true valuation functions, i.e., V is a DSE.*

Proof. Consider any relay node RN_i, any strategy W_i of RN_i, and any W_{-i}.

In order to compare $U_i(W_i, W_{-i})$ with $U_i(V_i, W_{-i})$, we distinguish two cases.

Case A: With both strategies W_i and V_i, the same value of x^* is computed in line 5 of the main algorithm. In this case,

$$x^*(V_i, W_{-i}) = x^*(W_i, W_{-i}).$$

Hence, from lines 7 and 8 we can easily see that

$$\forall j \neq i, F_j(V_i, W_{-i}) = F_j(W_i, W_{-i}).$$

Consequently,

$$F_i(V_i, W_{-i}) = R_R - \sum_{j \neq i}^{1 \leq j \leq N} F_j(V_i, W_{-i}) \tag{3.27}$$

$$= R_R - \sum_{j \neq i}^{1 \leq j \leq N} F_j(W_i, W_{-i}) \tag{3.28}$$

$$= F_i(W_i, W_{-i}). \tag{3.29}$$

Since the equation in line 11 of MICOR main algorithm is independent from W_i, the solution to the equation, x_i^* is also independent from W_i. In line 13 of MICOR main algorithm, P_i is computed using x_i^*, $\{F_j\}_{j=1}^N$ and other variables that are independent from W_i. Hence,

$$P_i(V_i, W_{-i}) = P_i(W_i, W_{-i}). \tag{3.30}$$

Combining Equations (3.29)(3.30), we get that

$$U_i(V_i, W_{-i}) = U_i(W_i, W_{-i}).$$

Case B: With strategies W_i and V_i, different values of x^* are computed in line 5 of MICOR main algorithm. In this case,

$$x^*(V_i, W_{-i}) \neq x^*(W_i, W_{-i}).$$

Therefore, $F_i(V_i, W_{-i})$ may not be equal to $F_i(W_i, W_{-i})$. We further distinguish two subcases.

Subcase B-1: $F_i(V_i, W_{-i}) \geq F_i(W_i, W_{-i})$. In this subcase, by Lemma 3, we

know that for all $j \neq i$, $F_j(V_i, W_{-i}) \leq F_j(W_i, W_{-i})$. Now we study the utility difference:

$$\begin{aligned}
& U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \\
= \quad & P_i(V_i, W_{-i}) - V_i(F_i(V_i, W_{-i})) - (P_i(W_i, W_{-i}) \\
& -V_i(F_i(W_i, W_{-i}))) \\
= \quad & P_i(V_i, W_{-i}) - P_i(W_i, W_{-i}) - (V_i(F_i(V_i, W_{-i})) \\
& -V_i(F_i(W_i, W_{-i}))) \\
= \quad & P_i(V_i, W_{-i}) - P_i(W_i, W_{-i}) - \int_{F_i(W_i, W_{-i})}^{F_i(V_i, W_{-i})} V_i'(f)\mathrm{d}f.
\end{aligned}$$

Since $V_i()$ is convex, V_i' is increasing. Consequently, for any f such that $F_i(W_i, W_{-i}) < f < F_i(V_i, W_{-i})$,

$$\begin{aligned}
V_i'(f) &< V_i'(F_i(V_i, W_{-i})) \\
&= x^*(V_i, W_{-i}).
\end{aligned}$$

Hence,

$$\begin{aligned}
& U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \\
\geq \quad & P_i(V_i, W_{-i}) - P_i(W_i, W_{-i}) \\
& -(F_i(V_i, W_{-i}) - F_i(W_i, W_{-i})) \cdot x^*(V_i, W_{-i}) \\
= \quad & (\textstyle\sum_{j \neq i}^{1 \leq j \leq N} (W_j(F_j(W_i, W_{-i})) - W_j(F_j(V_i, W_{-i})))) \\
& -(F_i(V_i, W_{-i}) - F_i(W_i, W_{-i})) \cdot x^*(V_i, W_{-i}) \\
= \quad & (\textstyle\sum_{j \neq i}^{1 \leq j \leq N} (W_j(F_j(W_i, W_{-i})) - W_j(F_j(V_i, W_{-i}))))
\end{aligned}$$

$$
\begin{aligned}
& - \sum_{j\neq i}^{1\leq j\leq N}(F_j(W_i, W_{-i}) - F_j(V_i, W_{-i})) \\
& \cdot x^*(V_i, W_{-i}) \\
= \quad & \sum_{j\neq i}^{1\leq j\leq N}(\tfrac{W_j(F_j(W_i,W_{-i}))-W_j(F_j(V_i,W_{-i}))}{F_j(W_i,W_{-i})-F_j(V_i,W_{-i})} \\
& -x^*(V_i, W_{-i})) \cdot (F_j(W_i, W_{-i}) - F_j(V_i, W_{-i})),
\end{aligned}
$$

where the first equality is due to line 13 of MICOR main algorithm and the fact that $x_i^\star$ is independent of W_i, and the second equality is due to the feasibility of the algorithm. Applying Lagrange mean-value theorem, we get that

$$
\begin{aligned}
& U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \\
\geq \quad & \sum_{j\neq i}^{1\leq j\leq N}((W_j'(\eta) - x^*(V_i, W_{-i})) \\
& \cdot (F_j(W_i, W_{-i}) - F_j(V_i, W_{-i}))),
\end{aligned}
$$

where $F_j(V_i, W_{-i}) < \eta < F_j(W_i, W_{-i})$. Since $W_j()$ is convex,

$$
W_j'(\eta) > W_j'(F_j(V_i, W_{-i})) = x^*(V_i, W_{-i}).
$$

Therefore,

$$
U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \geq 0.
$$

Subcase B-2: $F_i(V_i, W_{-i}) < F_i(W_i, W_{-i})$. In this subcase, by Lemma 3, we know that for all $j \neq i$, $F_j(V_i, W_{-i}) > F_j(W_i, W_{-i})$. Many steps in this subcase are similar to Subcase B-1 and thus we will skip part of them.

$$
\begin{aligned}
& U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \\
= \quad & \int_{F_i(V_i,W_{-i})}^{F_i(W_i,W_{-i})} V_i'(f)\mathrm{d}f - (P_i(W_i, W_{-i}) - P_i(V_i, W_{-i}))
\end{aligned}
$$

For any f such that $F_i(V_i, W_{-i}) < f < F_i(W_i, W_{-i})$,

$$\begin{aligned} V_i'(f) &> V_i'(F_i(V_i, W_{-i})) \\ &= x^*(V_i, W_{-i}). \end{aligned}$$

Hence,

$$\begin{aligned} & U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \\ \geq \quad & (F_i(W_i, W_{-i}) - F_i(V_i, W_{-i})) \cdot x^*(V_i, W_{-i}) \\ & -(P_i(W_i, W_{-i}) - P_i(V_i, W_{-i})) \\ = \quad & \textstyle\sum_{j\neq i}^{1\leq j\leq N} \big(\frac{W_j(F_j(V_i,W_{-i})) - W_j(F_j(W_i,W_{-i}))}{F_j(V_i,W_{-i}) - F_j(W_i,W_{-i})} \\ & -x^*(V_i, W_{-i})) \cdot (F_j(V_i, W_{-i}) - F_j(W_i, W_{-i})). \end{aligned}$$

Therefore, similar to Subcase B-1, we can get that

$$U_i(V_i, W_{-i}) - U_i(W_i, W_{-i}) \geq 0.$$

In summary, we always have that

$$U_i(V_i, W_{-i}) \geq U_i(W_i, W_{-i}),$$

and so V is a DSE. □

3.4.5 Channel Assignment to Reduce Monopoly

Recall that in the previous section, we assume channels have already been assigned appropriately. In this section, we study how to assign channels appropriately so that MICOR main algorithm can be executed to compute F_i and P_i.

By Proposition 10, the feasibility of MICOR main algorithm requires that there is no monopoly. Ideally, we should develop a channel assignment algorithm that can prevent monopoly. Nevertheless, we notice that whether there is a monopoly depends not only on the channel assignment, but also on R_R, the additional transmission rate the cooperative relay needs to provide.

As mentioned previously, there is no monopoly if the cooperative relay can be done with any relay node being excluded. When R_R gets large enough, it becomes simply impossible to satisfy the relay traffic demand with one relay node being excluded. (In fact, when R_R gets even larger, it also becomes impossible to satisfy the demand even with all the relay nodes in RNS.) Therefore, to characterize how "good" a channel assignment is in terms of monopoly reduction, we define a concept called monopoly threshold as follows.

Definition 11. *Given a network, for a channel assignment Z, the* monopoly threshold *is the maximum relay transmission rate that this network is able to provide while ensuring there is no monopoly. We write it as $MT(Z)$.*

In a network with a channel assignment Z, if the relay traffic demand $R_R \leq MT(Z)$, then we know under this channel assignment Z, our MICOR main algorithm is feasible. Otherwise, if $R_R > MT(Z)$, then there must be a monopoly.

On the other hand, when the channel assignment Z varies, the monopoly threshold $MT(Z)$ also varies. The greater $MT(Z)$ is, the better Z is (in terms of

reducing monopoly). The best possible channel assignment Z should maximize $MT(Z)$, in order to allow more relay traffic demand R_R. Formally, we have the following definition:

Definition 12. *A channel assignment $Z^\star$ achieves the* optimal monopoly reduction *if*

$$Z^\star = \arg\max_Z MT(Z).$$

Hence, it would be best if we could design a channel assignment algorithm that achieves the optimal monopoly reduction. Unfortunately, the problem of channel assignment is hard in general [90], and it seems even harder if we want to achieve the optimal monopoly reduction. Consequently, we design a distributed channel assignment algorithm that assigns the channels in a way that does not tend to generate monopoly. Our algorithm is heuristic and does not have a guarantee for optimal monopoly reduction. However, our experimental results in evaluation section show that it achieves the optimal monopoly reduction with a very good probability.

The main idea behind this algorithm is simple: To prevent relay nodes from becoming indispensable for the relay, we can assign the channels in a "fair" manner. We make sure no relay node gets significantly more channels than other relay nodes for relaying traffic.

Figure 3.19 describes our channel assignment algorithm for each potential relay node RN_i. In our algorithm, periodically RN_i searches with a probability for a new pair of channels (x, y), such that i can communicate with SN and DN through channels x and y respectively. This probability of search depends on n, the number of channel pairs that i has already obtained. The more channel

1. $n \leftarrow 0$;
2. $RCN \leftarrow \{1, 2, \ldots, K\}$;
3. for $m = 1$ to r
4. $\quad CP \leftarrow \{(x,y) | x, y \in RCN, x \neq y,$
$\qquad a_{N+1,x} a_{i,x} = 1, a_{i,y} a_{N+2,y} = 1\}$;
5. $\quad$ if $CP = \emptyset$
6. $\quad\quad$ go to step 8;
7. $\quad$ else if $rand < p_i(n)$
$\quad\quad$ find a pair of channels $(x,y) \in CP$;
$\quad\quad$ broadcast a claim for channels x and y;
$\quad\quad$ $n \leftarrow n + 1$;
$\quad\quad$ $RCN \leftarrow RCN \backslash \{x, y\}$;
8. $\quad$ wait for time Δt while listening
9. $\quad\quad$ upon receiving a claim for channels (x', y')
10. $\quad\quad\quad$ if $\{x, y\} \cap \{x', y'\} \neq \emptyset$
11. $\quad\quad\quad\quad$ $n \leftarrow n - 1$;
$\quad\quad\quad\quad$ $RCN \leftarrow RCN \cup \{x, y\}$;
$\quad\quad\quad\quad$ broadcast an announcement that
$\quad\quad\quad\quad$ it has released channels x and y;
12. $\quad\quad\quad$ else
13. $\quad\quad\quad\quad$ $RCN \leftarrow RCN \backslash \{x', y'\}$;
14. $\quad\quad$ upon receiving an announcement that
$\quad\quad$ x' and y' have been released
15. $\quad\quad\quad$ $RCN \leftarrow RCN \cup \{x', y'\}$;

Figure 3.19. Channel assignment algorithm for MICOR in Section 3.4.

pairs RN_i has already obtained, the less probability it has for searching for new channel resources. Formally, we use a decreasing function $p_i()$ to compute this probability from n.

Clearly, there may be a conflict between two or more relay nodes, if they try to obtain the same channel at the same time. To address this issue, we require that, once a node finds a pair of new channels through its search, the node needs to broadcast a claim for this pair of channels. After broadcasting the claim, the node should wait for time Δt. Here Δt is a parameter greater than 2 times the

maximum transmission time between two relay nodes. If RN_i receives a claim for either channel in the pair within Δt, RN_i should release both channels and announces the release to all other nodes. Otherwise, after the waiting period of Δt is completed, RN_i knows that it has obtained the pair of channels.

We repeat the above procedure for a number of times, and then with high probability, no more channel can be assigned for relay. In practice, the number of repetitions is pretty small and thus our channel assignment algorithm is very efficient (see evaluation section for experimental results).

One observation of this channel assignment algorithm is that the overhead is mainly determined by Δt, because the involved computation is relatively simple (see evaluatoin section for experimental results). Consequently, it is very important to make Δt small, although Δt should definitely be greater than twice the maximum transmission time.

Another observation is that we can re-execute our channel assignment algorithm multiple times, in order to achieve the optimal monopoly reduction with a better probability. Note that this repetition[8] of the entire algorithm is *different* from the repetition of channel searching inside the algorithm. When the entire repetition process finishes, each relay node has to follow the channel assignment which has the highest monopoly threshold. (If the highest monopoly threshold is achieved by more than one channel assignment, then all relay nodes follow the first one.) Our evaluation results demonstrate that the optimal channel assignment can be achieved with probability of over 99% when the algorithm is executed for 4 times.

[8]This repetiton of the entire algorithm needs synchronization among relay nodes. Such synchronization can be achieved using various methods, e.g., using synchronous clocks.

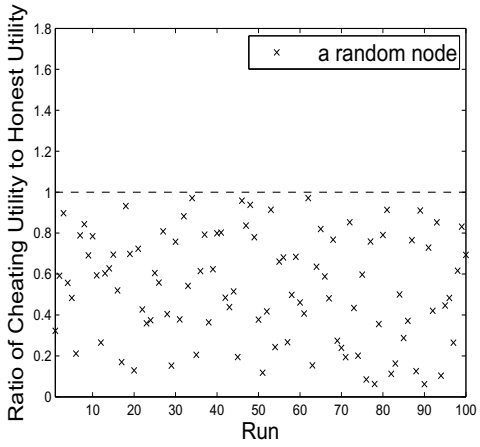

Figure 3.20. Ratio (of cheating utility to honest utility) measurement if $V(x) = k(\log c - \log(c - g(x)))$ is applied in Section 3.4.

3.4.6 Evaluations

We implement our MICOR algorithms and test them using NS2. In particular, we conduct three sets of experiments:

- First, we verify that, when MICOR is used, each relay node has strong incentives to report its true valuation function. To achieve this objective, we study how a node's strategy affects its utility under MICOR. Our results show that, it is always the best strategy for a node to report the true valuation function.

- Second, we evaluate the monopoly reduction of the MICOR channel assignment algorithm. Our results show that, when the MICOR channel assignment algorithm is executed for a few times (and the parameter r is set appropriately), with probability of more than 99%, the optimal monopoly reduction is achieved.

- Third, we measure the computation and communications overheads of MICOR. The results show that MICOR is very efficient.

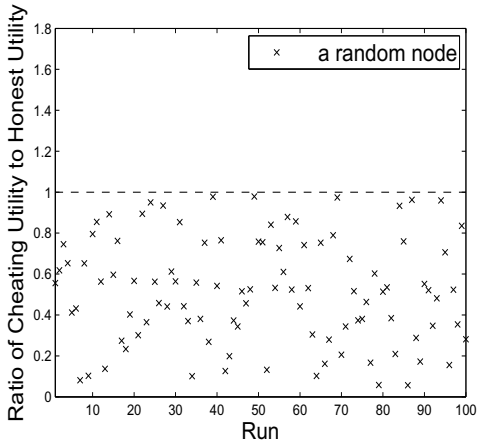

Figure 3.21. Ratio measurement if $V(x) = \frac{kx}{c-g(x)}$ is applied in Section 3.4.

Note we do not measure how our protocols can affect the network performance (e.g. overall throughput, etc), because ideally under our protocols the same improvement can be achieved as in [67, 68].

Relay Node's Utility

In the first set of experiments, we test how the strategy of a potential relay node affects its utility. In NS2, we simulate a wireless network with 10 nodes, and test three types of valuation functions of each relay node RN_i:

- $V_i(x) = k_i(\log c_i - \log(c_i - g(x)))$,
- $V_i(x) = \frac{k_i x}{c_i - g(x)}$,
- $V_i(x) = k_i(\frac{1}{(c_i-g(x))^2} - \frac{1}{c_i^2})$,

where

$$g(x) = \begin{cases} x & \text{if } 0 \leq x < c_i \\ c_i & \text{otherwise} \end{cases}, \tag{3.31}$$

and k_i is a parameter determined by node RN_i itself in each valuation function. In our experiments, we assume k_i is an integer such that $1 < k_i \leq 5$. It is

randomly picked for each RN_i before each run of our test. RN_i's maximum relay transmission rate c_i is also picked randomly from the range $0 < c_i < 10$.

It is easy to verify that the above defined utility functions all satisfy the five requirements (RQ1)-(RQ5) in system model. Recall that we have extended the definition of utility function and defined $V_i(x) = +\infty$ for all $x \geq c_i$.

We test utilities of nodes for each of the above valuation functions respectively. For each valuation function, we have 100 runs of our test. In all of these runs, we assume $R_R = 50$.

We choose a random node RN_X and observe its utility change over all the 100 runs. In each run, we first randomly pick a strategy for each of the other 9 nodes. This strategy could be either honestly reporting the true valuation function, or cheating by reporting a false valuation function. Then, we measure the utility of RN_X in two cases: In the first case, the strategy of RN_X is honest and reports its true valuation function V_X. In the second case, the strategy of RN_X cheats and reports a random false valuation function W_X. Note that all other nodes' strategies are the same in these two cases, with only RN_X's strategy being different. We observe how RN_X's utility changes as its strategy changes.

More precisely, we compute the utility ratio $\frac{U_X(W_X,W_{-X})}{U_X(V_X,W_{-X})}$. Here the numerator $U_X(W_X, W_{-X})$ is the utility of RN_X when RN_X chooses to cheat (and so we call it the *cheating utility*). The denominator $U_X(V_X, W_{-X})$ is the utility of RN_X when RN_X takes an honest strategy (and so we call it the *honest utility*). This ratio between the cheating utility and the honest utility reflects whether cheating benefits. If it is less than or equal to 1, then cheating does not benefit. If it is more than 1, then cheating benefits.

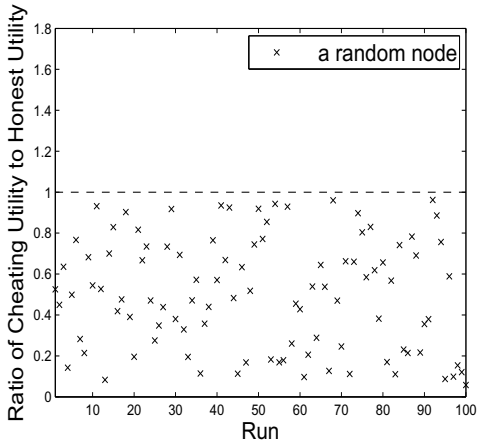

Figure 3.22. Ratio measurement if $V(x) = k(\frac{1}{(c-g(x))^2} - \frac{1}{c^2})$ is applied in Section 3.4.

In Figures 3.20-3.22, we can see the results of our tests, for the three valuation functions respectively. We notice that in all circumstances, the utility ratio is less than or equal to 1. Hence, cheating is never beneficial.

Monopoly Reduction

In the second set of experiments, we evaluate the monopoly reduction of our MICOR channel assignment algorithm. Specifically, we measure the percentages that the MICOR channel assignment algorithm achieves the optimal monopoly reduction in 6 different settings.

- Setting S1: $|RNS| = 3$, $K = 5$, and $r = 2$.
- Setting S2: $|RNS| = 3$, $K = 5$, and $r = 4$.
- Setting S3: $|RNS| = 5$, $K = 7$, and $r = 4$.
- Setting S4: $|RNS| = 5$, $K = 7$, and $r = 6$.
- Setting S5: $|RNS| = 10$, $K = 8$, and $r = 6$.
- Setting S6: $|RNS| = 10$, $K = 8$, and $r = 9$.

For each setting, we randomly simulate 100 networks and run the MICOR channel assignment algorithm in all these 100 networks.

In order to see whether the monopoly reduction is optimal, we also do an exhaustive search in each simulated network to find the channel assignment that has the optimal monopoly reduction. (Note that such exhaustive searches should not be used in practice because they require much more time than MICOR channel assignment algorithm. We use them here only for the purpose of comparison.) In each setting, we compare the computed channel assignment by our MICOR channel assignment algorithm with the channel assignment discovered by the exhaustive search algorithm. In this way, we obtain the percentage that the MICOR channel assignment algorithm achieves the optimal monopoly reduction.

In the entire set of experiments, we use $p_i(n) = 0.6 - 6n/5K$ for all relay nodes.

The results are shown in Figure 3.23. In all settings, the monopoly reduction gets better when we repeat the algorithm for more times. However, there is a clear difference between settings S1, S3, S5, and settings S2, S4, S6. In settings S1, S3, and S5, even if the algorithm is repeated for 8 times, the percentage of optimal monopoly reduction is still below 80%. In contrast, in settings S2, S4, S6, the monopoly reduction is optimal in all the 100 simulated networks, as long as we repeat the algorithm for 4 times or more.

The reason for this difference lies in that settings S1, S3, S5 have smaller values for r (2, 4, 6, respectively), while settings S2, S4, S6 have greater values for r (4, 6, 9, respectively) With r being too small, relay nodes may not get enough chance to obtain their channels. Hence, in practice, we should avoid using too

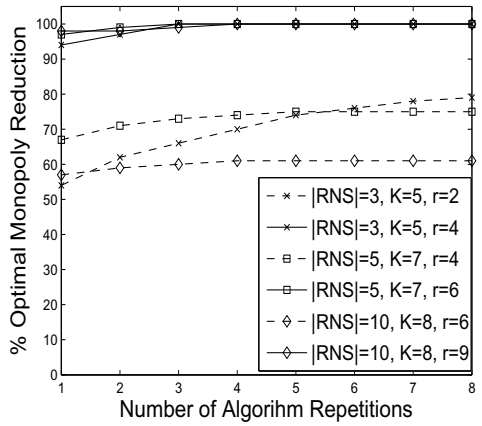

Figure 3.23. Percentage of optimal monopoly reduction that MICOR achieves in Section 3.4.

Table 3.3. Overheads of MICOR main algorithm in Section 3.4

$\|RNS\|$	Total Overhead	% Computation	% Communications
2	2.0ms	0.6%	99.4%
3	2.0ms	0.7%	99.3%
5	3.0ms	0.7%	99.3%
8	5.0ms	0.6%	99.4%
10	6.0ms	0.6%	99.4%

small values for r.

In summary, if we set r to a not-too-small value and repeat the MICOR channel assignment algorithm for a not-too-small number of times, then the optimal monopoly reduction can be achieved with probability close to 1. In situations similar to this set of experiments, it suffices to set $r = 9$ and repeat the MICOR channel assignment algorithm for at least 4 times.

Overhead

In the third set of experiments, we measure the computation and communications overheads of MICOR. The results on the MICOR main algorithm and the channel assignment algorithm are presented in Tables 3.3 and 3.4, respectively.

Table 3.3 shows the measured total overhead (including computation and communications) for the MICOR main algorithm, as well as what percentages

Table 3.4. Overheads of MICOR channel assignment algorithm in Section 3.4

r	Total Overhead	% Computation	% Communications
2	4.0ms	0.6%	99.4%
4	8.0ms	0.3%	99.7%
6	12.0ms	0.2%	99.8%
8	16.0ms	0.1%	99.9%
10	20.0ms	0.1%	99.9%

of this overhead being for computation and for communications, respectively. Clearly, when the number of relay nodes increases, the total overhead increases correspondingly. It is also clear that for all numbers of relay nodes we have experimented with, the total overhead is small (no more than 6.0 milliseconds).

For the MICOR main algorithm, the dominating overhead is for communications. In all of our experiments, the computation overhead can essentially be ignored when compared with the communications overhead.

Table 3.4 shows the measured total overhead for a *single* execution of the channel assignment algorithm, as well as what percentages of this overhead being for computation and for communications, respectively. The total overhead is linear in r. Even if $r = 10$, the total overhead of channel assignment is still small (20 milliseconds). Consequently, if the channel assignment algorithm is repeated for 4 times (in order to achieve the optimal monopoly reduction, as we discuss in the evaluation section), the total overhead becomes 80 milliseconds, which is still small.

Similar to the MICOR main algorithm, the channel assignment algorithm spends nearly all time on communications.

3.4.7 Summary

In this work we design a payment-based system MICOR, which provides incentives to cooperative relay among secondary users. MICOR guarantees that

nodes have strong incentives to be honest when reporting their valuation functions, so that the assignment of relay traffic and computing of payment can be carried out correctly. MICOR is efficient in that it has low overheads for computation and communications.

To support the MICOR main algorithm, we also design a channel assignment algorithm for MICOR, which achieves the optimal monopoly reduction with a very good probability. An open problem is whether it is possible to design a channel algorithm that always achieves the optimal monopoly reduction.

Bibliography

[1] A. Shamir. How to Share a Secret. In *CACM, VOL. 22, NO. 11*, 1979.

[2] T. Elgamal. A Public Key Cryptosystem and a Signature Scheme based on Discrete Logarithms. In *IEEE Transactions on Information Theory, VOL. IT-31*, JULY 1985.

[3] Y. Desmedt and Y. Frankel. Threshold Cryptosystems. *In Advances in Cryptology C Crypto*, 1990.

[4] L. Zhou and Z. J. Haas. Securing Ad Hoc Networks. *IEEE Network Magazine 13(6)*, 1999.

[5] S. Capkun, L. Buttyan, and J. P. Hubaux. Self-Organized Public-Key Management for Mobile Ad Hoc Networks. *IEEE Transactions on Mobile Computing 2(1)*, 2003.

[6] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: A taxonomy and some efficient constructions. In *Proceedings of IEEE INFOCOM*, 1999.

[7] V. Shoup. Practical Threshold Signatures. In *Advances in Cryptology - EUROCRYPT*, 2000.

[8] L. Zhou, F. B. Schneider and R. V. Renesse. COCA: A Secure Distributed Online Certification Authority. *ACM Transactions on Computer Systems, Vol. 20, No. 4, November 2002.*

[9] R. Canetti, S. Halevi, J. Katz. A Forward-Secure Public-Key Encryption Scheme. In *Advances in CryptologyłEUROCRYPT*, 2003.

[10] O. Goldreich. Secure Multi-Party Computation. *Manuscript, 1998*

[11] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network Information Flow. *IEEE Transactions on Information Theory, VOL. 46, NO. 4*, JULY 2000.

[12] T. Ho, R. Koetter, M. Medard, D. R. Karger and M. Effros. The Benefits of Coding over Routing in a Randomized Setting. *IEEE International Symposium on Information Theory*, 2003.

[13] S. Biswas and R. Morris. ExOR:Opportunistic Multi-Hop Routing for Wireless Networks. In *Proceedings of ACM SIGCOMM*, 2005.

[14] C. Gkantsidis and P. Rodriguez. Network Coding for Large Scale Content Distribution. In *Proceedings of IEEE INFOCOM*, 2005.

[15] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen. Polynomial time algorithms for multicast network code construction. In *IEEE Transactions on Information Theory, VOL. 51, NO. 6*, JUNE 2005.

[16] A. Jiang. Network Coding for Joint Storage and Transmission with Minimum Cost. In *IEEE International Symposium on Information Theory*, 2006.

[17] S. Chachulski, M. Jennings, S. Katti and D. Katabi. Trading Structure for Randomness in Wireless Opportunistic Routing. In *Proceedings of ACM SIGCOMM*, 2007.

[18] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A High-Throughput Path Metric for Multi-hop Wireless Routing. In *Wireless Networks 11*, 2005.

[19] John C. Platt, Nello Cristianini, John Shawe-Taylor. Large Margin DAGs for Multiclass Classification In *Advances in Neural Information Processing Systems*, MIT Press.

[20] S. Bhadra, S. Shakkottai, and P. Gupta. Min-cost selfish multicast with network coding. In *Information Theory, IEEE Transactions on*, vol. 52, no. 11, pp. 5077-5087, 2006.

[21] X. Zhang and B. Li. Dice: a game theoretic framework for wireless multipath network coding. In *Proceedings of ACM MobiHoc*, Hong Kong, China, May 2008.

[22] Z. Li. Cross-monotonic multicast. In *Proceedings of IEEE INFOCOM*, Phoenix, AZ, Apr. 2008.

[23] X. Liang. Matrix games in the multicast networks: maximum information flows with network switching. In *IEEE Trans. on Information Theory*, vol. 52, pp. 2433-2466, June 2006.

[24] F. Wu, T. Chen, S. Zhong, L. E. Li, and Y. R. Yang. Incentive Compatible Opportunistic Routing for Wireless Networks. In *Proceedings of ACM MOBICOM* San Francisco, Sep. 2008

[25] S. Zhong, J. Chen, Y. R. Yang. Sprite: a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *Proceedings of IEEE InfoCom'03*, 1987-1997, March 2003.

[26] M. Mahmoud and X. Shen. ESIP: Secure Incentive Protocol with Limited Use of Public-Key Cryptography for Multi-hop Wireless Networks. In *IEEE Transactions on Mobile Computing (IEEE TMC)*, vol.10, no.7, pp.997-1010, July 2011.

[27] C. Cortes and V. Vapnik. Support-Vector Networks. In *Machine Learning*, 20, 1995.

[28] J. Price and T. Javidi. Network coding games with unicast flows. In *IEEE J. on Selected Areas in Commun.*, vol. 26, Sept. 2008.

[29] C. Schuldt, I. Laptev, B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR, Proceedings of the 17th International Conference on*, 2004.

[30] D. Chen, H. Bourlard, J.P. Thiran. Text identification in complex background using SVM. In *CVPR, Proceedings of the IEEE Computer Society Conference on*, 2001.

[31] C. G. M. Snoek, M. Worring, A. W. M. Smeulders. Early versus late fusion in semantic video analysis. In *MULTIMEDIA '05 Proceedings of the 13th annual ACM international conference on Multimedia*

[32] K. Flouri, B. Beferull-Lozano, and P. Tsakalides. Distributed consensus algorithms for SVM training in wireless sensor networks. In *16th European Signal Processing Conference*, Lausanne, Switzerland, August 25-29, 2008

[33] Y. Zhang, W. Lee and Y. Huang. Intrusion Detection Techniques for Mobile Wireless Networks. In *WIRELESS NETWORKS*, Volume 9, Number 5, 545-556, 2003

[34] M. Brunato, R. Battiti. Statistical learning theory for location fingerprinting in wireless LANs. In *Computer Networks*, Volume 47, Issue 6, 22 April 2005, Pages 825-845

[35] R. Jain, D.M. Chiu, W. Hawe. A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems. In *DEC Research Report TR-301*, 1984.

[36] W. Jakes. Microwave Mobile Communications. *IEEE Press*, NJ, 1994.

[37] UCLA Mobile Systems Laboratory. Global Mobile Information Systems Simulation Library. http://pcl.cs.ucla.edu/projects/glomosim/

[38] L. Giupponi and C. Ibars. Bayesian potential games to model cooperation for cognitive radios with incomplete information. In *Proceedings of IEEE International Conference on Communications*, Jun. 2009.

[39] L. Harasim, S. R. Hiltz, L. Teles, M. Turoff. Learning Networks: A Field Guide to Teaching and Learning On-Line. *The MIT Press*, 1995.

[40] Austin Bond. Learning music online - An accessible learning program for isolated students. In *NCVER*, 2003. 37 p.

[41] Sarah Granger. The Simplest Security: A Guide To Better Password Practices. *http://www.symantec.com/connect/articles/simplest-security-guide-better-password-practices*.

[42] E.V. Belmega, B. Djeumou, and S. Lasaulce. What happens when cognitive terminals compete for a relaying node? In *Proceedings of ICASSP*, 2609 - 2612, Apr. 2009.

[43] A. Brandenburger. Nash Equilibrium: Existence. Technical report available at http://pages.stern.nyu.edu/ abranden/nashproof-01-04-07.pdf.

[44] M. Jakobsson, J. P. Hubaux, and L. Buttyan. A micro-payment scheme encouraging collaboration in multi-hop cellular networks. In *Proceedings of Financial Crypto*, 15 - 33, Jan. 2003.

[45] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of ACM MOBICOM*, 255 - 265, Aug. 2000.

[46] P. Michiardi, R. Molva. Game theoretic analysis of security in mobile ad hoc networks. *Research Report No.02-070, Institute Eurecom*, 2002.

[47] P. Michiardi, R. Molva. CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proceedings of the Communication and Multimedia Security Conference*, Sept. 2002.

[48] F. Wu, T. Chen, S. Zhong, L. E. Li, and Y. R. Yang. Incentive-compatible opportunistic routing for wireless networks. In *Proceedings of ACM MOBICOM*, 303 - 314, Sept. 2008.

[49] N. Ben Salem, L. Buttyan, J. P. Hubaux, and M. Jakobsson. A charging and rewarding scheme for packet forwarding in multi-hop cellular networks. In *Proceedings of ACM MOBIHOC*, 13 - 24, Jun. 2003.

[50] S. Buchegger and J. Y. L. Boudec. Performance analysis of the CONFIDANT protocol: Cooperation of nodes fairness In dynamic Ad-hoc networks. In *Proceedings of ACM MOBIHOC*, 226 - 236, Jun. 2002.

[51] T. Chen and S. Zhong. INPAC: An enforceable incentive scheme for wireless networks using network coding. In *Proceedings of IEEE INFOCOM*, Mar. 2010.

[52] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. IEEE Transactions on Information Theory, 46(4):1204 - 1216, 2000.

[53] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen. Polynomial time algorithms for multicast network code construction. IEEE Transactions on Information Theory, 51(6):1973 - 1982, 2005.

[54] R. Koetter and M. Mdard. An algebraic approach to network coding. IEEE/ACM Transactions on Networking, 11(5):782 - 795, 2003.

[55] S. R. Li, R. W. Yeung, and N. Cai. Linear network coding. IEEE Transactions on Information Theory, 49(2):371 - 381, 2003.

[56] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein. Growth codes: Maximizing sensor network data persistence. In Proceedings of ACM SIGCOMM06, Pisa, Italy, Sept. 2006.

[57] S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: Analog network coding. In Proceedings of ACM SIGCOMM07, Kyoto, Japan, Aug. 2007.

[58] S. Katti, D. Katabi, W. Hu, H. S. Rahul, and M. Mdard. The importance of being opportunistic: Practical network coding for wireless environments. In Proceedings of the 43rd Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept. 2005.

[59] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Mdard, and J. Crowcroft. XORs in the air: Practical wireless network coding. In Proceedings of ACM SIGCOMM06, Pisa, Italy, Sept. 2006.

[60] D. S. Lun, N. Ratnakar, R. Koetter, M. Mdard, and a. H. L. E. Ahmed. Achieving minimum-cost multicast: A decentralized approach based on network coding. In Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM05), Miami, FL, Mar. 2005.

[61] Yi Tang, Lijie Wang, David Grace, Jibo Wei. Utility based cooperative spectrum leasing in cognitive radio networks. In *Wireless Communication Systems (ISWCS), 2012 International Symposium on*, Aug. 2012.

[62] J. Mitola III and G. Maguire Jr. Cognitive radio: making software radios more personal. In *IEEE Personal Communications*, Vol. 6, No. 4, Pages 13-18, 1999.

[63] S. Haykin. Cognitive radio: brain-empowered wireless communications. In *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 2, Pages 201-220, 2005.

[64] I. Akyildiz, W. Lee, M. Vuran, and S. Mohanty. Next generation/dynamic

spectrum access/cognitive radio wireless networks: A survey. In *Computer Networks*, Vol. 50, No. 13, Pages 2127-2159, 2006.

[65] Y. Zou, J. Zhu, B. Zheng, Y. Yao. An adaptive cooperation diversity scheme with best-relay selection in cognitive radio networks. In *IEEE Transactions on Signal Processing*, Vol. 58, No. 10, Pages 5438-5445, Oct. 2010.

[66] W. Su, J.D. Matyjas, S. Batalama. Active cooperation between primary users and cognitive radio users in cognitive ad-hoc networks. In *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing*, Pages 3174-3177, Mar. 2010.

[67] J. Jia, J. Zhang, and Q. Zhang. Cooperative relay for cognitive radio networks. In *Proceedings of IEEE INFOCOM*, Pages 2304-2312, April 2009.

[68] Q. Zhang, J. Jia, and J. Zhang. Cooperative relay to improve diversity in cognitive radio networks. In *IEEE Communications Magazine*, Vol. 47, No. 2, Pages 111-117, Feb. 2009.

[69] G. Zhao, C. Yang, G.Y. Li, D. Li, A.C.K. Soong. Power and channel allocation for cooperative relay in cognitive radio networks. In *IEEE Journal of Selected Topics in Signal Processing*, Vol. 5, No. 1, Pages 151-159, Feb. 2011.

[70] J. Liu, W. Chen, Z. Cao, Y. Zhang. Cooperative beamforming aided incremental relaying in cognitive radios. In *IEEE International Conference on Communications (ICC)*, Pages 1-5, June 2011.

[71] L. Li, X. Zhou, H. Xu, G.Y, Li, D. Wang, A. Soong. Simplified relay selection and power allocation in cooperative cognitive radio systems. In *IEEE*

Transactions on Wireless Communications, Vol. 10, No. 1, Pages 33-36, Jan. 2011.

[72] B. Lamparter, K. Paul, and D. Westhoff. Charging support for ad hoc stub networks. In *Computer Communications*, Vol. 26, Pages 1504-1514, 2003.

[73] L. Anderegg and S. Eidenbenz. Ad hoc-VCG: A truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *Proceedings of ACM MOBICOM*, Pages 245-259, Sep. 2003.

[74] W. Wan, X. Y. Li, and Y. Wang. Truthful multicast routing in selfish wireless networks. In *Proceedings of ACM MOBICOM*, Sep. 2004.

[75] S. Zhong, L. E. Li, Y. G. Liu, and Y. R. Yang. On designing incentive-compatible routing and forwarding protocols in wireless ad hoc networks. In *Proceedings of ACM MOBICOM*, Pages 117-131, Aug. 2005.

[76] W. Wang, S. Eidenbez, Y. Wang, and X. Y. Li. OURS - Optimal unicast routing systems in non-cooperative wireless networks. In *Proceedings of ACM MOBICOM*, Pages 402-413, Sep. 2006.

[77] M. Felegyhazi, M. Cagalj, S. Saeedi Bidokhti, and J.P. Hubaux. Non-cooperative multi-radio channel allocation in wireless networks. In *Proceedings of IEEE INFOCOM*, May 2007.

[78] F. Wu, S. Zhong, and C. Qiao. Globally optimal channel assignment for non-cooperative wireless networks. In *Proceedings of IEEE INFOCOM*, April 2008.

[79] L. Gao and X. Wang. A game approach for multi-channel allocation in multi-hop wireless networks. In *Proceedings of ACM MOBIHOC*, May 2008.

[80] R.D. Vallam, A. Kanagasabapathy, C. Siva Murthy. A non-cooperative game-theoretic approach to channel assignment in multi-channel multi-radio wireless networks. In *ACM Journal of Wireless Networks*, Vol. 17, No. 2, Feb. 2011.

[81] O. Simeone, I. Stanojev, S. Savazzi, Y. Bar-Ness, U. Spagnolini, and R. Pickholtz. Spectrum leasing to cooperating secondary ad hoc networks. *IEEE Journal on Selected Areas in Communications*, Vol 26, 203 - 213, Jan. 2008.

[82] J. Zhang, Q. Zhang. Stackelberg game for utility-based cooperative cognitive radio networks. In *Proceedings of ACM MOBIHOC*, 23 - 32, May 2009.

[83] H. Yao and S. Zhong. Towards cheat-proof cooperative relay for cognitive radio networks. In *Proceedings of ACM MOBIHOC*, May 2011.

[84] Z. Guan, T. Melodia, D. Yuan, and D. A. Pados. Distributed spectrum management and relay selection in interference-limited cooperative wireless networks. In *Proceedings of ACM MOBICOM*, Sep. 2011.

[85] N. Shastry and R. Adve. Stimulating cooperative diversity in wireless ad hoc networks through pricing. In *IEEE Proceedings of ICC*, Pages 3747-3752, 2006.

[86] B. Wang, Z. Han, and K. Liu. Distributed relay selection and power control for multiuser cooperative communication networks using buyer/seller game. In *Proceedings of IEEE INFOCOM*, Pages 544-552, 2007.

[87] J. Huang, Z. Han, M. Chiang, and H. Poor. Auction-based resource allocation for cooperative communications. In *IEEE Journal on Selected Areas in Communications*, Vol. 26, No. 7, Pages 1226-1237, Sep. 2008.

[88] I. Stanojev, O. Simeone, U. Spagnolini, Y. Bar-Ness, R.L. Pickholtz. Cooperative ARQ via auction-based spectrum leasing. In *IEEE Transactions on Communications*, Vol. 58, No. 6, Pages 1843-1856, June 2010.

[89] Y. Xiao, G. Bi, D. Niyato. Game theoretic analysis for spectrum sharing with multi-hop relaying In *IEEE Transactions on Wireless Communications*, Vol. 10, No. 5, Pages 1527-1537, May 2011.

[90] W. Si, S. Selvakennedy, A.Y. Zomaya. An overview of channel assignment methods for multi-radio multi-channel wireless mesh networks. In *Journal of Parallel and Distributed Computing*, Vol. 70, No. 5, Pages 505-524, May 2010.