OPPORTUNISTIC COOPERATION IN WIRELESS NETWORKS

HU ZHENGQING

NATIONAL UNIVERSITY OF SINGAPORE

2010

OPPORTUNISTIC COOPERATION IN WIRELESS NETWORKS

HU ZHENGQING
(*B.Eng. (Hons), NUS*)

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2010

Name	:	Hu Zhengqing
Degree	:	Doctor of Philosophy
Supervisor(s)	:	Prof. Tham Chen-Khong
Department	:	Department of Electrical & Computer Engineering
Thesis Title	:	OPPORTUNISTIC COOPERATION IN WIRELESS NETWORKS

Abstract

Cooperation plays a fundamental role in wireless networks. Many cooperative techniques, such as cooperative diversity, MIMO, and opportunistic routing have been designed and implemented on real networks. However, due to the dynamics of the wireless network, and the lack of information, in many cases, there are only some uncertain opportunities of cooperation. Techniques designed for these cases are known as opportunistic cooperation techniques. Two important questions needed to be answered, about these techniques, are: 1) when to cooperate and 2) whom to cooperate with. Other challenges faced by such techniques are "on the fly" decision making, overhead minimization, and etc. In this thesis, these issues are studied in the field of Wireless LANs and Wireless Sensor Networks by applications.

In the area of Wireless LANs, throughput is one of, if not the most, important performance metric. After exploring the opportunity of cooperation in the MAC layer, we propose a new MAC protocol. This is CCMAC, a coordinated cooperative MAC for wireless LANs. It is designed to improve the throughput performance in the region near the AP (a bottleneck area), through cooperative communication. The most unique feature is that, it can coordinate nodes to perform concurrent transmissions, when the opportunities are found. Through analysis and simulation, we show that CCMAC can significantly shorten the transmission time for wireless stations with low data rate link to the AP. It has better throughput performance than other MAC protocols, such as CoopMAC and legacy IEEE 802.11.

In the area of wireless sensor networks (WSN), traditional network rout-

ing algorithms can be challenged by nodes' propensities to go to sleep, move around, or even break down. It is costly in terms of communication and energy consumption for routing information to be kept up-to-date. Based on the idea of geographic opportunistic forwarding, we propose a new hybrid opportunistic forwarding protocol: Geographic Multi-hop-Sift (GMS), which combines two opportunistic forwarding techniques: priority list and random access. It is designed to be both energy efficient and robust against channel fluctuation or frequent changes of network topology. In this protocol the next hop relay node is selected by neighboring nodes themselves, using a Sift "game". Meanwhile, the sender node can optionally influence the selection process, based on the list of preferred nodes (LPN).

Lastly, a general coordination scheme, based on priority list technique, is proposed. Normally, the overhead caused by coordination is non-negligible for an opportunistic cooperation. The proposed scheme takes both the overhead and the potential benefits into consideration. Based on this scheme, an algorithm with polynomial time complexity is given, to find the best priority list, which can optimize the user-defined metrics.

Keywords : Cooperation, Algorithm design, MAC protocol, Opportunistic Routing

Acknowledgment

I would like to give my heartfelt thanks to my supervisor, Prof. Tham Chen Khong, for his guidance, support and encouragement throughout my study.

I would also like to thank my parents and my wife. They always give me their unconditional love and support.

Last, but not least, I want to thank my friends and colleagues in CNDS lab for their kind assistance and suggestions on research and other issues. The interesting discussion during lunch and coffee time is so enjoyable.

July 12, 2010

Contents

List of Figures

Chapter 1

Introduction

Wireless communication technology has brought fundamental changes to data networking, and is making integrated networks a reality. By freeing the user from the cord, personal communications networks, wireless LAN's, mobile radio networks and cellular systems, provide a way of fully distributed mobile computing and communications, anytime, anywhere.

However, such flexibility comes with some unique constraints of wireless communication. For example, the signal attenuation in wireless communication is significantly higher than in normal wired communication; due to the broadcast nature of wireless transmission, a wireless transmission may cause a large amount of interference to other wireless stations, using the same channel, nearby; and wireless channels are often unstable and hard to estimate, etc.

These constraints bring many negative effects to wireless networks, for example the unstable connectivity, low data rate, etc. Moreover, such effects are very hard for each individual node to combat. Hence, recently,

researchers have found that cooperation plays a fundamental role in wireless networks.

Cooperation is the process of working or acting together, which can be accomplished by both intentional and non-intentional agents. By gathering resources from different agents, many times cooperative strategies can achieve better performance than non-cooperative strategies. For example, to combat the severe signal attenuation, instead of pushing the transmitting energy at the sender end, data can be relayed by multi-hop forwarding. Secondly, to fight with the constraint of unstable and unreliable wireless channel, the MIMO techniques have been developed. They use the idea of spatial diversity, by having multiple senders and multiple receivers, the overall channel gain can be much higher and more stable. Furthermore, when other constraints, like power/energy consumption, QoS (quality of service), are to be taken into consideration, cooperation may become more important in order to meet the application's requirements.

Although cooperation is very useful in wireless communication, when applying this kind of techniques into wireless networks, we often face one challenging problem: the dynamics of the networks. Since nodes may move around, channels are unstable, if cooperation is blindly applied, it may just create extra cost without bringing any benefits. Sometimes, it may even lead to a worse result compared with not applying cooperation, e.g. sometimes direct transmission is better than multi-hop relaying. Furthermore, without proper information about the networks, it is even harder to select the cooperation partners. Hence, in many situations, we need to determine

whether to use a cooperative strategy and with whom to cooperate, on the fly. It means, these decisions need to be made with partial information. These cooperation strategies are studied in this thesis, which is known as opportunistic (on demand) cooperation.

1.1 Challenges

There are several challenges when designing and implementing the opportunistic cooperation.

Firstly, we need to identify the application, which is suitable for opportunistic cooperation techniques. Opportunistic cooperation techniques are not free. Most of them come with overhead such as additional signal transmission and energy consumption. In addition, many applications can be implemented without opportunistic cooperation. Hence, we need to know beforehand, whether implementing such techniques is really beneficial for the given applications.

Secondly, we need information acquisition and online decision making strategies. By the nature of the opportunistic cooperation, it can perform well only if it has enough and correct information about the network status, which includes the status information about other cooperation partners and the environment. However, due to the dynamics of the networks, it may be costly to have the complete knowledge. Hence, we need to balance between the amount of information gathered and the cost incurred. Simultaneously, opportunistic cooperation needs a good online decision-making strategy, which can make decisions adaptively with the current knowledge.

For example, it needs to decide quickly of whether the cooperation can be started, or aborted, or wait for more information to come.

Lastly, we need to have coordination among the nodes involved, especially, about the coordination of message passing. Since opportunistic cooperation choose members of the the cooperation, on the fly, many nodes may be involved. This may lead to many signals/messages exchanging, and the contentions of wireless channel among nodes. Hence, coordination among these nodes plays a very important role. A good coordination helps to minimize the packet collision, and more importantly, selects good cooperation partners efficiently.

1.2 Related Work

In this section, we introduce two successful examples of applying opportunistic cooperation in wireless networks. They are cooperative diversity and opportunistic forwarding/routing. Both examples contain all the challenges mentioned above. Hence, the approaches of the given examples provide good hints on how to deal with opportunistic cooperation in other applications.

1.2.1 Cooperative Diversity

This is a cooperative multiple antenna technique which exploits user diversity by decoding the combined signal of the relayed signal and the direct signal in wireless multi-hop networks. A conventional single hop system uses direct transmission where a receiver decodes the information only based on the direct signal while regarding the relayed signal as interference, whereas

the cooperative diversity considers the other signal as contribution. That is, cooperative diversity decodes the information from the combination of two signals. Hence, it can be seen that cooperative diversity is an antenna diversity that uses distributed antennas belonging to each node in a wireless network.

There are three basic relaying strategies in cooperative diversity: Amplify-and-Forward, Decode-and-Forward and Compress-and-Forward. In [1], Laneman et al. introduced the schemes Amplify-and-Forward, Decode-and-Forward, and a hybrid scheme that switches between these two. Amplify-and-Forward is non-regenerative, i.e., the helper does not extract data from the signal received from the source. The signal is amplified and relayed to the destination. In contrast to this non-regenerative relaying, with Decode-and-Forward the data is regenerated at the helper. After receiving the signal, both helpers extract symbols which are demodulated to code words and decode these code words to data bits. These bits are re-encoded and retransmitted to the destination. Here, the partner's data can be checked for errors, e.g., by using Cyclic Redundancy Check (CRC), prior to the relaying, and more powerful codes may be employed. The Compress-and-Forward cooperative relaying strategy was initially suggested in Theorem 6 of [2]. This scheme strikes a balance between the regenerative and non-regenerative methods. On one hand, the received signal is only demodulated to digital symbols instead of being decoded to bits. On the other hand, these symbols are not directly repeated as a signal in phase 2. In order to reduce redundancy, the symbols are compressed and included in the relayed packet.

Recently, many researchers applied the idea of cooperative diversity in real network. For example, [3] enhances slotted ALOHA with cooperative relaying and evaluates its performance gains. The articles [4], [5], and [6] propose modifications to IEEE 802.11 networks with the cooperative relaying extension. All of these protocols use the opportunistic cooperation strategies, which exchanges information between neighboring nodes and select the relay nodes on the spot. More details of these protocols are introduced in chapter 3. Results have shown that the throughput performance have been improved by using these protocols.

1.2.2 Opportunistic Routing

Opportunistic routing is another excellent application of opportunistic cooperation. We know that, routing protocols for wireless networks have traditionally focused on finding the "best" path to forward packets between the source and destination. However, such approaches are vulnerable to node or link failures, which commonly happen in wireless networks. As a result, although such algorithms are relatively simple, it may not be the best approach in many kinds of wireless networks, such as wireless sensor networks (WSN), wireless mesh networks (WMN), etc.

One alternative approach, which is known as the opportunistic routing, chooses the routing path "on the fly". By having multiple relay candidates at each hop, it tries to choose the best and currently available nodes among them. Hence, it improves the performance when some expected or unexpected failure happens. Many opportunistic routing protocols have been

proposed such as [7], [8], [9], [10], [11], [12], [13].

1.3 Contributions and Thesis overview

Motivated from the previous work, the following topics are studied in this thesis:

- design and implementation of two opportunistic cooperative relaying protocols in IEEE 802.11 wireless LAN, which can further improve the throughput performance by scheduling concurrent transmissions based on cooperation;
- design and implementation of a new opportunistic forwarding protocol in WSN, which produces better performance with less overhead;
- design of a general coordination scheme, which takes overhead into consideration and can optimize user defined metrics.

The thesis is organized as follows:

In chapter 2, we give an overview of Markov Decision Process (MDP) and graph theory, both of which are well-studied models with very unique properties. We are particularly interested in the problem of finite-state MDP and partially observable Markov decision process in the field of MDP; and the problem of weighted vertex coloring and maximum independent set in the field of graph theory. These mathematical models will be used later in real networking problems.

In chapter 3, we explore the benefits of cooperative communication and concurrent transmissions at the medium access control (MAC) layer in IEEE

802.11 WLAN uplink. A novel coordinated cooperative MAC (CCMAC) protocol is proposed, which couples both strategies to improve the throughput performance. CCMAC has different transmission modes. One of the modes will be chosen during packet transmission, based on the channel condition and the helper's status. Through analysis and simulation, we verified that CCMAC can achieve substantial throughput performance improvement, without incurring significant network overheads, in the uplink (from clients to the access point) of Wireless LAN.

In chapter 4, the idea of cooperation communication and concurrent transmission is extended to the downlink of IEEE 802.11 WLAN. A sender initiated concurrent cooperative MAC (SI-CCMAC) protocol is proposed. Unlike the uplink, this time the access point is the initiator, which has more information about the networks than normal clients. However, it also brings new challenges such as fairness and long-term optimization, which we will explain in detail in chapter 4. SI-CCMAC copes with these new conditions well. Similar to CCMAC, SI-CCMAC can achieve substantial throughput performance improvement, without incurring significant network overheads, in the downlink of Wireless LAN.

In chapter 5, a novel hybrid opportunistic forwarding protocol for wireless sensor networks, which we refer to as the Geographic Multi-hop Sift (GMS) protocol, is proposed. The important feature of GMS is that it seamlessly combines a centralized coordination scheme with a distributed coordination scheme. By doing this, it improves the efficiency whilst being robust to link or node failures. In addition, it is able to overcome the

problems encountered by other similar schemes such as high probability of packet collisions and periodic information exchanges.

In chapter 6, a general coordination scheme to select a node to perform opportunistic cooperation, based on a priority list technique, is proposed. Unlike the normal priority list schemes, which usually give the cooperation partner with higher future benefit a higher priority; the proposed scheme takes both the overhead and future benefits into consideration. Based on this scheme, an algorithm with polynomial time complexity is given, to find the best priority list, which can optimize user-defined metrics.

Finally, chapter 7 concludes the whole thesis.

Chapter 2

Theoretical models

Before we talk about the application of opportunistic cooperation in wireless networks, we are going to introduce two useful theoretical models in this chapter. They are Markov Decision Process (MDP) and graph theory. These models are frequently adopted to solve network problems in the real world. We are also going to adopt these models in the opportunistic cooperation algorithms, proposed in this thesis.

2.1 Markov Decision Process

Markov Decision Process is a mathematical framework for modeling decision-making in a stochastic environment, where outcomes are random but under the influence of the decision maker. MDPs were introduced by Bellman (1957) [14]. Today they are used in a variety of areas, including robotics, automated control, and economics, for modeling a wide range of optimization problems.

Markov decision processes are an extension of Markov chains; the difference is the addition of actions and rewards. If there were only one action, or if the action to take (policy) were fixed for each state, a Markov decision process would reduce to a Markov chain.

More precisely a Markov Decision Process is a discrete time stochastic control process characterized by components (S, A, P, R), where S is a finite set of states, A is a finite set of actions, P: $S \times A \times S \leq 1$ defines a probabilistic transition model given the current state and action to the next state, and R: $S \times A \rightarrow \Re$ defines the reward function of choosing an action under a specific state. The common objective is normally to find the action in each state, which maximizes the expected discounted reward, represented as $\sum_{t=1}^{\infty} \gamma^{t-1} r_t$, where r_t is the immediate reward received at time t, and $\gamma \in (0, 1)$ is a discount factor.

The solution to a Markov Decision Process can be expressed as a policy π: $S \rightarrow A$, a probability function of choosing actions under given states. Note that once an MDP is associated with a fixed policy, *i.e.* the probability of choosing actions for each state is fixed, then the MDP behaves like a Markov chain.

The standard family of algorithms to calculate the policy requires storage for two arrays indexed by the state: value V, which contains real values, and policy π which contains actions. At the end of the algorithm, π will contain the solution and V will contain the discounted sum of the rewards to be

earned (on average) by following that solution.

$$\pi(s) = \arg\max_a (R_a(s, s') + \sum_{s'} P_a(s, s')V(s')) \tag{2.1}$$

$$V(s) = R(s) + \gamma \sum_{s'} P_{\pi(s)}(s, s')V(s') \tag{2.2}$$

The algorithm then has the following two kinds of steps, which are repeated in some order for all the states until no further changes take place. Generally, there are two common approaches, which are *value iteration* [14] and *policy iteration* [15] to solve different problems.

2.1.1 Partially observable Markov decision process

A Partially Observable Markov Decision Process (POMDP) is a generalization of a Markov Decision Process. In a POMDP model, the system dynamics are determined by an MDP. However, the decision maker cannot directly observe the underlying state. Instead, it must infer a distribution over the state based on a model of the world and some local observations.

The POMDP framework is a more realistic model for many real problems, compared with MDP. Applications of POMDP include robot navigation problems, machine maintenance, and planning under uncertainty in general. However, this extension of MDP dramatically increases the complexity, which makes exact solutions intractable. In order to act optimally, an agent may need to take into account all the previous history of observations and actions, instead of just the current state it is in. Formally, a POMDP contains an underlying MDP, plus an observation space O and an observation function Z. In an MDP, the agent has full knowledge of the

system state, therefore, $S \equiv O$. In a POMDP, determining the current state, becomes problematic. The reason is that the same observation may be observed in different states. Hence, we have a new stochastic mapping function Z, where Z: $S \times A \times O$, which specifies the relationship between system states and observations. $Z(\acute{s}, a, \acute{o})$ is the probability that an agent is in state $\acute{s}$ after observing $\acute{o}$ and executing action a. Formally, a POMDP is a tuple of (S, A, P, R, O, Z).

The standard approaches for solving MDPs are value iteration and policy iteration. However, in the case of POMDP, exact methods for solving POMDPs are intractable, in part because optimal policies can be either very large, or even infinite. For example, in exact policy iteration, the number of controller nodes may grow exponentially in the horizon length. In value iteration, the number of vectors required to represent the value function multiplies at a doubly exponential rate.

One of the approximation techniques is therefore to restrict the set of policies. The goal is then to find the best policy within that restricted set. Since all policies can be represented as (possibly infinite) policy graphs, a widely used restriction is to limit the set of policies to those representable by finite policy graphs, or finite-state controllers (FSC), of some bounded size. This allows us to achieve a compromise between the requirement that courses of action should depend on certain aspects of observable history, and the ability to control the complexity of the policy space.

2.2 Graph Theory

Graph theory is the general term of the study of graphs, which gives mathematical structures used to model pairwise relations between objects from a certain collection. A "graph" in this context refers to a collection of vertices or "nodes" and a collection of edges that connect pairs of vertices. A graph may be undirected, which means all the edges in the graph has no direction; or otherwise directed. A graph structure can be extended by assigning a weight to each edge of the graph. Graphs with weights, or weighted graphs, are used to represent structures in which pairwise connections have some numerical values. For example if a graph represents a road network, the weights could represent the length of each road.

Structures that can be represented as graphs are ubiquitous, and many problems of practical interest can be represented by graphs. The link structure of a network could be represented by a directed graph: the vertices are the network stations and a directed edge from station A to station B exists if and only if A can send data to B directly. A similar approach can be taken to problems in travel, biology, computer chip design, and many other fields. The development of algorithms to handle graphs is therefore of major interest in computer science, especially the computer networks field.

There are many interesting problems included in the content of graph theory, such as subgraph problem, graph coloring problem, network flow problem, etc. In this section, we have particular interest in two problems, which are vertex coloring problem and maximum independent set problem.

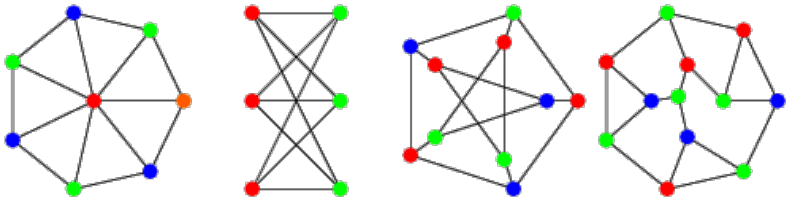


Figure 2.1: Some proper vertex colorings of some graphs.

2.2.1 Vertex coloring problem

A vertex coloring problem is the simplest form of a graph coloring problem, it is a way of coloring the vertexes of a graph such that no two adjacent vertexes share the same color. Examples of some of the proper vertex coloring are shown in Figure 2.1.

Other coloring problems can be transformed into a vertex version. For example, an edge coloring of a graph is just a vertex coloring of its line graph, and a face coloring of a planar graph is just a vertex coloring of its planar dual.

A weighted vertex coloring problem is an extension of vertex coloring problem. This time, a weight w, which is a positive integer number, is assigned to each vertex. Then, each vertex is required to be colored by at least w colors, and there are no two adjacent vertexes sharing the same color. A weighted vertex coloring problem can be convert to a normal vertex coloring problem by substituting each vertex as a **complete graph** with w nodes, where w is the weight of each node.

A coloring using at most k colors is called a (proper) k-coloring. The smallest number of colors needed to color a graph G is called its chromatic number. A graph that can be assigned a (proper) k-coloring is k-colorable,

and it is k-chromatic if its chromatic number is exactly k. Vertex coloring is computationally hard. It is NP-complete to decide if a given graph admits a k-coloring for given k except for the cases $k = 1$ and $k = 2$. Especially, it is NP-hard to compute the chromatic number [16].

The common approaches to solve a vertex coloring problem are Brute-force search [17],[18], [19]. However, due to the high computational complexity of the exact methods, algorithms based on heuristics are frequently used. Generally, these heuristics methods can be grouped with the following types: **construction Heuristics** [20], [21]; **local search methods** [22], [23]; **hybrid metaheuristics** [24], [25], etc.

2.2.2 Maximum independent set

In graph theory, an independent set of a graph, is a set of vertex, which none of them are connected directly in the graph. A maximal independent set is an independent set that is not a subset of any other independent set. A maximal independent set is also a dominating set in the graph, and every dominating set that is independent must be maximal independent, so maximal independent sets are also called independent dominating sets. A graph may have many maximal independent sets of widely varying sizes; a largest maximal independent set is called a maximum independent set (MIS). Figure 2.2 shows some examples of finding the maximum independent set in a graph. Nodes selected in the MIS are shown in red color.

If S is a maximum independent set in some graph, it is a maximum clique or maximum complete subgraph in the complementary graph. A maximum

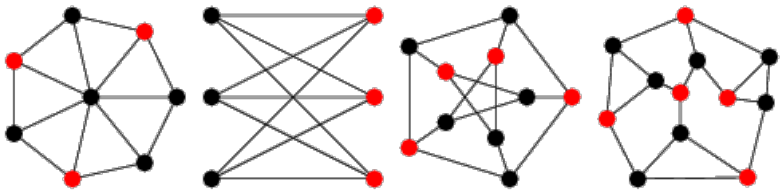


Figure 2.2: The maximum independent sets of some graphs.

clique is a set of vertexes that induces a complete subgraph, and that is not a subset of the vertexes of any larger complete subgraph. That is, it is a set S such that every pair of vertexes in S is connected by an edge and every vertex not in S is missing an edge to at least one vertex in S. Hence, finding the maximum independent set is equivalent as finding the maximum clique.

The maximum independent set problem is important for applications in Computer Science, Operation Research, and Engineering. There are many applications of the MIS such as graph coloring, wireless channel assignment, register allocation for a compiler.

It is well known that, finding the maximum independent set of a general graph is a NP-hard problem [16]. Algorithms to find the exact solution, such as [26] and [27], often have the complexity growing exponentially with the number of vertex. Hence, heuristics, like [28] are often used to approximately solve this problem when the graph is large.

The phrase "maximal independent set" is also used to describe maximal subsets of independent elements in mathematical structures other than graphs, and in particular in vector spaces and matroids.

2.3 Conclusion

Two of the well-designed mathematical models (MDP and graph theory), together with their common solutions, have been introduced in this chapter. Both models have special properties and widely adapted in various networking problems. In the following chapters, we will demonstrate how to apply these theoretical models in our research, which is about the opportunistic cooperation.

Chapter 3

Concurrent Cooperative MAC (uplink)

In wireless LANs, throughput is one of, if not the most, important performance metric. This metric becomes more critical at the bottleneck area of the network, which is normally the area around the Access Point (AP). In this chapter, we propose CCMAC, a coordinated cooperative MAC for wireless LANs. It is designed to improve the throughput performance in the region near the AP through cooperative communication, where data is forwarded through a two-hop high data-rate link instead of a one-hop low data-rate link. The most unique feature is that, it can coordinate nodes to perform concurrent transmissions which can further increase throughput. To optimize the performance, the coordination problem is formulated as a POMDP (Partially Observable Markov Decision Process) and solved by a Reinforcement Learning (RL) algorithm. Through analysis and simulation, we show that CCMAC can significantly shorten the transmission time for

stations with low data rate links to the AP and CCMAC has better throughput performance than other MAC protocols, such as CoopMAC and legacy IEEE 802.11.

3.1 Introduction

IEEE 802.11 (WiFi) based wireless LANs have become extremely popular in the past decade. One of the main reasons for the success is that WiFi provides a high data rate communication medium with low cost. According to the standards, IEEE 802.11b supports data rates up to 11 Mbps; IEEE 802.11a and 802.11g support data rates up to 54 Mbps; the recently approved IEEE 802.11n draft 3.0 [29] supports data rates up to 248 Mbps. However, in the real world, it may be more important to consider the achievable data rate, i.e. throughput. This is because noise and interference, together with signal loss due to path loss and fading, may severely reduce the achievable data rate from its theoretical maximum value.

To mitigate some of the above mentioned problems, techniques referred to as cooperative communication are being developed. Due to the broadcast nature of the wireless medium, wireless stations can overhear the transmissions from their neighboring stations. Utilizing this property, the key idea of cooperative communication is to let the intermediate wireless stations, known as relay stations, process the overheard signal and retransmit them to the destination. The destination combines the signals received from the source and the relay stations, and hence, may get a more accurate message by reducing the effects of path loss and fading.

Researchers have started to adopt cooperative communication techniques in various wireless communication networks, including cellular, ad hoc and mesh networks. In WLAN, algorithms using this technique can be found in [5], [6], and [30]. In most of the existing work, the decode-and-forward relay strategy is practical and widely used, compared with other strategies such as amplify-and-forward, which requires expensive hardware circuits. The decode-and-forward strategy allows the relay node to relay the data after the sender's transmission, which is directly implementable on most existing hardwares.

In a network system, the overall throughput performance is usually limited by bottleneck links or a bottleneck area. In a wireless LAN, when there is only one access point (AP), the bottleneck of the network is normally at the region near the AP, which we shall call the "near-AP" region. This means that, even in a multi-hop wireless LAN, the overall throughput performance largely depends on the performance at the near-AP region. Hence, events in this region should be carefully considered.

One of our observations is that, when cooperative communication is applied, concurrent transmissions becomes possible even within the one-hop region of the AP, for transmissions to the AP. This means that, we can let multiple coordinated nodes transmit simultaneously, which can further increase the achievable throughput. The problem then becomes how to maximize the throughput through intelligent coordination. Our solution, described in Section 3.5, is that, by modeling the underlying problem as a Partially Observable Markov Decision Process (POMDP), we can use a Re-

inforcement Learning (RL) algorithm to coordinate the senders and optimize the throughput performance.

In this chapter, we propose a novel coordinated cooperative MAC (CCMAC) protocol. It can intelligently apply cooperative transmission, by two-hop relaying, and coordinate up to five concurrent transmissions within this region. To the best of our knowledge, this work is the first one which considers the coordination of concurrent transmission upon a random access MAC layer. We evaluate the performance of CCMAC by analysis and simulation, and show that CCMAC can reduce the transmission time, and hence, increase the throughput performance, for nodes with unfavorable direct channels to the AP. It outperforms the legacy IEEE 802.11 protocol and other relay-enabled MAC protocols, like CoopMAC [5].

3.2 IEEE 802.11 and Related Work

3.2.1 IEEE 802.11 (WiFi) Protocol

The IEEE 802.11 standard provides multi-rate wireless transmission capability through the use of different modulation schemes. For example, IEEE 802.11b supports rates of 1, 2, 5.5 and 11 Mbps, while IEEE 802.11a/g support rates of 6, 9, 12, 18, 24, 36, 48, and 54 Mbps. This means that WiFi has the capability to choose an appropriate data rate based on the prevailing channel condition.

There are two modes of the MAC protocol operation in WiFi. One is the point coordination function (PCF), and the other is the distributed coordination function (DCF). Between them, the DCF is used more widely. The

standard DCF protocol is described in [31]. It employs a carrier sense multiple access with collision avoidance (CSMA/CA) mechanism. Each wireless station can initiate a transmission after sensing that the channel is clear for a time period of a distributed inter-frame space (DIFS). However, packet collision may still occur at the receiver even if the channel is sensed to be clear by the senders. This is the well-known hidden terminal problem. To solve this problem, the RTS-CTS handshaking, which was first designed in MACA [32] and modified in MACAW [33], can also be employed in WiFi. The sender sends an RTS, and the receiver sends a CTS, to reserve the channel. Any other node, which overhears either of these packets, extracts the information of the channel reservation duration and updates its network allocation vector (NAV). This vector tells how long the node should keep silent. Although the RTS-CTS handshaking solves the hidden terminal problem, it creates additional overhead. In IEEE 802.11, the RTS-CTS mechanism is applied only when the data packet is larger than a certain threshold.

3.2.2 Related Work

The Auto Rate Fallback (ARF) protocol [34] is the first proposed algorithm to utilize the multi-rate capability of IEEE 802.11. In ARF, the sender chooses a higher data rate based on the history and falls back to a lower rate if several consecutive transmission failures happen. Later, the Receiver-Based Auto Rate (RBAR) protocol [35] was proposed. In RBAR, the receiver measures the SNR (signal-to-noise ratio) of the RTS packet. Based

on this SNR, the receiver tells the sender which modulation scheme to use. Since RBAR measures the channel quality just before the data transmission, it can choose the appropriate modulation scheme more accurately.

To apply the idea of cooperative communication in wireless LANs, the author of [36] proposed the Relay-Enabled PCF (rPCF) protocol. It employs a two-hop relaying mechanism in the PCF mode of WiFi, when the transmission time of this mechanism is shorter than the direct transmission. Two relay-enabled MAC protocols, rDCF [6] and CoopMAC [5], have also been proposed for the DCF mode of WiFi. These two protocols are very similar. Their basic idea is to minimize the transmission time by a two-hop relaying mechanism. In rDCF and CoopMAC, each sender maintains a list of helper nodes and decides which helper node should be chosen. In addition, they employ a similar handshaking sequence between the sender, helper and receiver by the control packets: RTS/HTS (helper-to-send)/CTS/ACK. The proposed CCMAC also uses this mechanism. However, the main difference between CCMAC and these two protocols is that, rDCF and CoopMAC do not consider the possibility of concurrent transmissions. In CCMAC, the capability of concurrent transmissions further increases the throughput performance in the near-AP region.

3.3 Motivation

We have mentioned above that, in a single AP wireless LAN, the bottleneck of the network is normally at the one-hop region of the AP. CCMAC is specially designed to operate in this region. It employs two techniques:

cooperative transmission and concurrent transmissions. In this section, we explain why these techniques can help increase the throughput performance and why we implement the techniques in the MAC layer. For simplicity, transmission overhead is not considered in this section. However, it will be considered later, in the protocol design and analysis sections.

3.3.1 Advantages of cooperative transmission in wireless LANs

In the presence of poor channel conditions, nodes in wireless LANs may only achieve a much lower data rate compared to the theoretical maximum value. For example, in an IEEE 802.11b wireless LAN, as shown in Figure 3.1, suppose the data rate of direct transmission from the source node, S_{s1}, to the destination node, AP, is R_{sd}. This R_{sd} is much lower than the maximum rate 11 Mbps. If one bit of data is to be transmitted directly from node S_{s1} to AP, the transmission time required is: $\frac{1}{R_{sd}}$. Suppose there is a helper node, S_{h1}, which has good wireless channels to both S_{s1} and AP. By using two-hop relaying, i.e. the source node sends data to the helper node, then the helper node relays the data to the destination node, it may actually shorten the transmission time. To illustrate that, suppose the transmission rate from the source node to the helper node is R_{sh} and the rate from the helper to the destination node is R_{hd}. The total time to transmit a bit of data from source to destination with the two-hop relaying is $\frac{1}{R_{sh}} + \frac{1}{R_{hd}}$. Hence, as long as equation (3.1) below is satisfied, for example, $R_{sh} = R_{hd} = 11$ Mbps and $R_{sd} = 2$ Mbps, two-hop relaying will have a better throughput performance.

$$\frac{1}{R_{sh}} + \frac{1}{R_{hd}} < \frac{1}{R_{sd}} \tag{3.1}$$

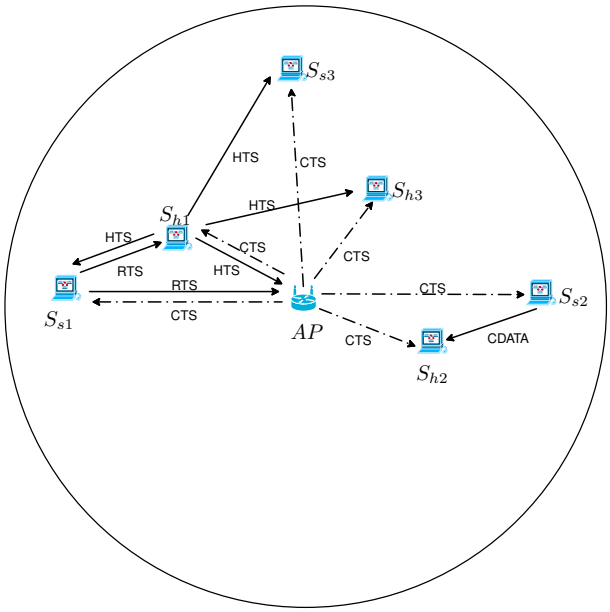


Figure 3.1: Network topology with seven nodes and the flow of messages.

3.3.2 Advantages of concurrent transmissions in Wireless LANs

In the uplink direction (traffic from user nodes to AP) of a wireless LAN, there may be multiple source nodes and multiple helper nodes, but there is only one destination node, which is the AP. In the AP's one-hop region, because of the single destination (AP), concurrent transmissions are impossible when direct transmission is applied. However, when two-hop relaying is implemented, concurrent transmissions become possible. The basic idea of the proposed concurrent transmissions scheme is that, multiple senders send data to their helpers simultaneously; after that, helpers relay the data to the destination (AP) one by one. For example, in Figure 3.1, node S_{s1} and

node S_{s2} can transmit to node S_{h1} and node S_{h2} at the same time. Then, node S_{h1} and S_{h2} can relay the data to the AP one after another. By using concurrent transmissions, the total transmission time can be reduced, thus, increasing the achievable throughput. More generally, suppose there are n senders which can perform the two-hop transmission simultaneously (later we prove in Theorem 3.1 that, $n \leq 5$ under certain assumptions). Suppose, the transmission rate from the i^{th} sender to the i^{th} helper is $R_{i,sh}$; the transmission rate from the i^{th} helper to the destination is $R_{i,hd}$. If every sender sends a bit of data to the destination, the total transmission time T_C for the concurrent transmissions scheme is shown in equation (3.2), where the first term is due to concurrent transmissions from sources to helper nodes, and the second term is due to one-by-one transmissions from helper nodes to the destination node. Under the same conditions, the total transmission time T_{nonC} for the non-concurrent two-hop transmission is shown in equation (3.3). Clearly, $T_C \leq T_{nonC}$. From these equations, we can see that two-hop concurrent transmissions can achieve even higher throughput than a two-hop non-concurrent transmission.

$$T_C = \max_i \frac{1}{R_{i,sh}} + \sum_{i=1}^{n} \frac{1}{R_{i,hd}} \tag{3.2}$$

$$T_{nonC} = \sum_{i=1}^{n} \frac{1}{R_{i,sh}} + \sum_{i=1}^{n} \frac{1}{R_{i,hd}} \tag{3.3}$$

3.3.3 MAC Layer versus Network Layer

The techniques of cooperative communication and concurrent transmissions can be performed through MAC layer relaying or network layer forwarding.

In CCMAC, MAC layer relaying is chosen. Some of the reasons, which have been discussed in [6], are: MAC layer relaying has a shorter delay and lower overhead compared to network layer forwarding, There is one additional reason which is specially related to concurrent transmissions. Network forwarding supports fewer concurrent transmissions than MAC layer relaying. Since each network forwarding needs an RTS-CTS handshake for itself, it may prevent other transmissions, even if they can be held concurrently. For example, in Figure 3.1, suppose the station S_{h2} can hear the message sent by station S_{h1}. When station S_{s1} is sending data to station S_{h1}, station S_{h2} receives the RTS request from station S_{s2}. This time, station S_{h2} will reject the request, because station S_{h1} has reserved the time using a CTS message. However, in CCMAC, both of the transmissions can still take place concurrently.

3.4 CCMAC Protocol

The CCMAC protocol is a contention-based random access MAC protocol for nodes in the one-hop region of the AP, including the AP. In this section, we first introduce techniques for data rate detection, helper selection and packet shaping, which are required before the real packet transmission. Then, we introduce how cooperation is achieved among the sources, helpers and destinations, and how the AP coordinates and enables concurrent transmissions from sources to helpers. Finally, we discuss a few issues related to the CCMAC protocol.

3.4.1 Transmission Rate Detection and Helper Selection

Recall from equation (3.1) that, the sender needs to know the three transmission rates among the sender, helper and AP before it can decide which transmission mode, i.e. direct transmission or two-hop transmission, to be applied. Before the transmission, the sender uses the cached information, i.e. the history, to make its decision. In the real transmission, similar to the RBAR [35], the sender chooses the data rate based on the detected signal to noise ratio (SNR). More precisely, when a sender joins the network, the transmission rate between the sender and AP, R_{sa}, is measured by the AP. Then, the AP will notify the sender about the value of R_{sa} in the CTS. The rate between sender and each helper, R_{sh}, is measured by overhearing the packets sent by the helper. The sender gets an estimation of R_{hs}, and uses it as R_{hs}. The sender detects that rate between each helper to the AP, R_{hd}, by overhearing the transmissions from helper to AP, from which the data rate information is extracted.

Once the sender has all the values described above, it picks the candidate helper nodes. The criteria for selecting the candidates is based on equation (3.4), which is an extension of equation (3.1). The difference between the two equations is that, in equation (3.4), the overheads are taken into consideration.

$$\frac{L}{R_{sh}} + \frac{L}{R_{hd}} + T_{HTS} + T_{overhead} < \frac{L}{R_{sd}} \tag{3.4}$$

In equation (3.4), L is the length of the data packet; the T_{HTS} is the time

to send the HTS using the base rate, e.g. 1 Mbps for 802.11b; $T_{overhead} = 2*(T_{SIFS}+T_{HD}+T_{PD}+T_{PH})$, where T_{HD} and T_{PD} are the hardware circuit delay and propagation delay respectively; T_{SIFS} is the SIFS duration, and T_{PH} is the delay caused by preamble header, including the PLCP header. .

Once the sender finds any of its neighbors satisfying equation (3.4), it will put the neighbor's ID into a *helper-table*. Similar to the *coopTable* in CoopMAC and *relay-table* in rDCF, the helper-table maintains information about the node ID, R_{sh} and R_{hd} for each helper candidate. In addition, a variable called *credit* is also saved and updated for each helper candidate.

A simple rule, with low computational cost, is applied to update the value of *credit*. The credit of every possible helper has an initial value 0.5 and varies between 0 and 1. Once a successful two hop transmission is completed, the credit of the selected helper will be increased by 0.1. Otherwise, if the transmission failed, the credit of the corresponding node will be decreased by 0.1. Once the credit fo a node equals 0, it will be deleted from the table and frozen for T minutes (in our implementation, the T equals 3). This means that, such nodes are not allowed to join this sender's helper-table for T minutes. After the frozen session, the sender will restart the rate detection session for that node. This algorithm is designed to deal with channel and node failures.

To select the helper from the helper-table, the sender will first consider the effective transmission time of each node. It is calculated by $\frac{1}{R_{sh}} + \frac{1}{R_{ha}}$, and the node with the smallest value will be selected as the helper. If two or more nodes have the same smallest value, the one with the higher credit

will be selected. If their credit are also the same, the node with the smallest ID number will be selected.

3.4.2 Packet Shaping

As illustrated in equation (3.2), if every node takes the same transmission time in the first hop (from sender to helper), concurrent transmissions will provide the most advantage over non-concurrent transmissions, since, every node has fully utilized the shared transmission time. However, since the transmission rate $R_{i,sh}$ is not a variable under control, full utilization rarely happens.

Thus, in order to achieve full utilization, a packet shaping technique is proposed. We know that nodes send data in packets. Suppose L_i is the packet size of node i, the first hop transmission time for node i is $\frac{L_i}{R_{i,sh}}$. Based on the IEEE 802.11 standard [31], variable packet sizes ($0 - 2312$ bytes of payload) are supported, which means L_i is an adjustable value. By shaping the packet size L proportional to the transmission rate R, we can let $\frac{L_1}{R_{1,sh}} = \frac{L_2}{R_{2,sh}} = \cdots = \frac{L_N}{R_{N,sh}}$. Hence, full utilization can be achieved. In CCMAC, a reference packet size, which is suggested to equal the maximum transmission unit (MTU) is defined. This reference packet size is associated with the sender node, which can achieve the maximum transmission rate, e.g. 11 Mbps in IEEE 802.11b, at the sender-helper link. Nodes, of which the packet size can be determined accordingly, with a lower transmission rate. For example, suppose the MTU is set as 1 KB. If $R_{i,sh}$ is 11 Mbps, then L_i is set as 1 KB. If $R_{j,sh}$ is 5.5Mbps, then L_j is set as 512Bytes.

3.4.3 The five different roles

Since CCMAC employs both cooperative communication and coordinated concurrent transmissions, it is more complicated than a normal random access MAC protocol. For example, in some cases, there can be 11 nodes involved at the same time. Based on our protocol, these nodes can be divided into five different groups (roles), and each node should act based on the current role it is playing.

The first role is the **main sender**. This role is played by the node that won the contention, which means, at any moment, there is one and only one node playing this role. The main sender's packet is sent towards the AP, which should be protected. Any other transmission which may interrupt the main sender's transmission should be prohibited. The algorithm details for the main sender is shown in Algorithm 1. The second role is the **helper of the main sender (main helper)**, which is selected by the main sender, based on the main sender's helper table. Hence, there can be either none or one node playing this role. The algorithm details for the main helper is shown in Algorithm 2. The third role is the **coordinator**, which is always played by the AP and is in charge of the coordination. It needs to know which nodes can send their packets without interfering with the ongoing transmissions (the data transmission of the main sender plus the nodes which have been selected as the TUs, see below) in order to maximize the throughput. Hence, the throughput is maximized. The details of how the coordination is learned is presented in next section. The logic of the AP, without the learning part, is shown in Algorithm 3. The fourth role is the

Algorithm 1: The sending process for the sender

```
Sender sends RTS, which contains the ID of the relay nodes, then
wait for HTS and CTS.
if CTS received then
    if HTS received then
        if data is cached by relay node then
            Apply half mode; wait.
        else
            Apply enhanced mode; send data to helper.
        end
    else
        Apply basic mode; send data to AP.
    end
    Wait for Ack
    if Ack received then
        Packet transmission success
    else
        Packet transmission failure, resend later.
    end
else
    Packet transmission failure, resend later.
end
```

Algorithm 2: The relaying process for the helper

```
After receiving the RTS from sender
Check whether the corresponding data packet was cached.
Send HTS, which contains the information of the data rate and
whether the data packet is cached.
if data is cached then
    Wait for CTS
    Send data packet to AP
else
    Wait for data packet from sender
    Send the received data to AP
end
```

Algorithm 3: The process of the AP

```
Wait for the RTS from sender
if relay node ID is specified then
    Wait a time period, during which the relay node sends the HTS.
    if HTS received then
        if data cached by helper then
            Calculate the corresponding NAV period.
            Send CTS and indicate it is a half mode transmission
        else
            Specify the list of TUs in the CTS.
            Calculate the corresponding NAV period.
            Send CTS and indicate it is an enhanced mode
            transmission
        end
    else
        Calculate the corresponding NAV period.
        Send CTS and indicate it is a basic mode transmission
    end
else
    Calculate the corresponding NAV period.
    Send CTS and indicate it is a basic mode transmission
end
Wait for the data packet.
if data is received then
    Send ACK
else
    Time out, back to idle.
end
```

time utilizer (TU). As will be proved in Theorem 3.1 later, there can be at most 4 TUs during one transmission. It means that the number of TUs can be from zero to four. The TUs are the nodes selected by the AP, which are can be listed in the AP's CTS. TUs try to utilize the sender's transmission time by sending their packets, known as the cached data (CDATA), to their helpers. However, even if a node is selected as a TU, it may not send its packet. This happens if the TU hears the HTS packet sent by the main helper. This is to avoid the packet collision and serves to protect the packet sent from the main sender.

The last role is the **helper of the time utilizer (HTU)**. For one TU, there is a corresponding HTU chosen by the TU. Hence, the number of HTUs is always the same as the number of TUs. During the transmission process, the HTUs always keep silent and they are just responsible to cache the data packets for the TUs. In CCMAC, each HTU only caches one packet for each TU. This means that, when a new packet is sent from the TU, it will replace any existing packet in the HTU's buffer. However, it will not affect the HTU's buffer of other TUs.

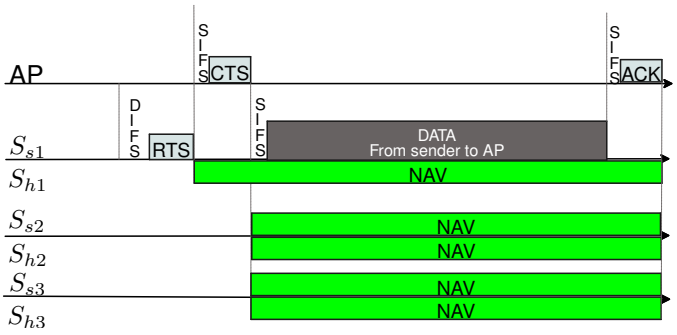
3.4.4 The three transmission modes

In CCMAC, the roles involved are determined by the current transmission mode. There are three different transmission modes for CCMAC, which are the **basic mode**, **enhanced mode** and **half mode**. To better illustrate the transmission process for the different modes, let us consider a sample network scenario. Suppose, there are seven wireless stations, one AP and

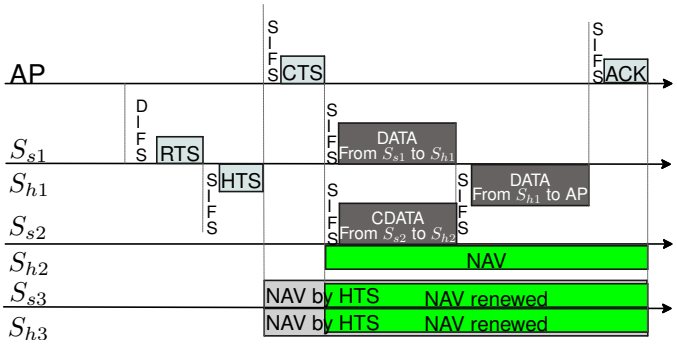
six client stations. Station S_{s1} wins the contention and becomes the main sender (refer to Figure 3.1 for the network topology).

The first transmission mode, known as the **basic mode**, happens when the sender does not need a helper or there is no helper which can help. This time, normal WiFi transmission will be used. RTS/CTS messages are exchanged before the data packets, and the data packets are sent directly to the AP. Figure 3.2(a) shows the data flow sequence for the basic mode. The second transmission mode, known as the **enhanced mode**, happens when the sender finds a helper node and the helper node does not cache the packet the sender is going to send. This time, a two-hop transmission is triggered. At the same time, the AP will decide which nodes can be selected as the TUs. Suppose, a node is selected as TU and does not hear the HTS from the main helper. It will send a packet to its helper with a tag number attached. Later, in the **half mode**, this tag number is to verify whether the packet is the correct packet to relay. Figure 3.2(b) shows the data flow sequence for the enhanced mode. Note that, station S_{s3} has been selected as TU but forced to keep silent, because it had heard the HTS. This shows how the transmission of the main sender is protected.

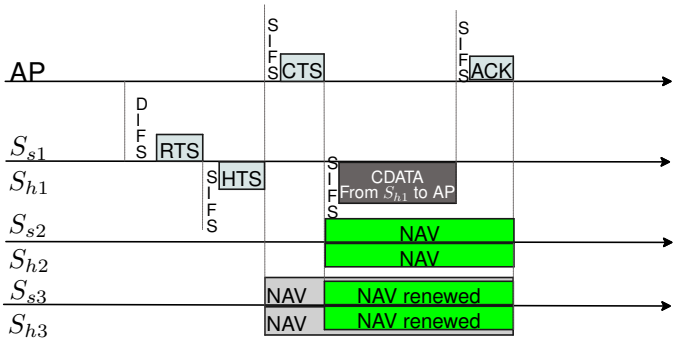
The last transmission mode, known as the **half mode**, happens when sender has a helper and the helper has cached the data the sender wants to send. Verification of the data is done by checking whether the tag number of the cached packet is the same as the tag number written in the RTS. If it does not match, the **enhanced mode** is applied. Otherwise, the **half mode** starts. This time, the helper will send the HTS and indicate that



(a) Message flow for the basic mode.



(b) Message flow for the enhanced mode.



(c) Message flow for the half mode.

Figure 3.2: The three different transmission modes.

the data has been cached. The transmission duration of the half mode is significantly shorter than the other two modes. This is because the data only needs to be transmitted once and through a fast link. Figure 3.2(c) shows the data flow sequence for the half mode.

To give a more detailed description of the algorithm, the pseudo code is presented for the main sender (**Algorithm 1**), helper (**Algorithm 2**) and AP (**Algorithm 3**) for the three different modes. We omit the pseudo codes for the TU and HTU, as they are fairly straightforward.

3.4.5 Discussions

Willingness of cooperation

In wireless community networks, like ad-hoc networks and mesh networks, we normally assume that client nodes are selfish. This means that nodes are only interested in their own benefits and care nothing about the overall network performance. As a result, nodes may not want to cooperate if the cooperations only favor other nodes but not themselves, e.g. relaying data for other nodes in an ad-hoc network. Hence, in such cases, cooperation has to be enforced through other means, which may not guarantee that the problem is solved and may induce extra cost to the network. In the case of CCMAC, where self throughput is the main interest for each node, the willingness of cooperation is not a problem. This is because when a sender is sending a packet, its helper cannot utilize this time period. Even if the helper does not help, it cannot obtain any additional throughput anyway. On the other hand, if the helper helps the sender, it saves the transmission

time for the sender. Such saved time will be accumulated and shared by all nodes in the network. Hence, the helper can also get more transmission time later. The only concern for the helper may be its energy consumption. However, it has been shown in [37] that, the energy-per-bit experienced by the helper stations is sometimes decreased by participating in cooperation. This is due to the reduction in idle energy consumption incurred by the helper. Whereas, in non-cooperative case, the helper takes longer time to wait and listen to the channel while a slow node is occupying the channel. Hence, our conclusion is that, even if all nodes are selfish, they should still be willing to use the CCMAC protocol and cooperate accordingly.

Implementation on multi-hop networks

Although the CCMAC protocol is designed for nodes in the near-AP region, it is still compatible with the WiFi protocol, which is the basic mode of CCMAC. Hence, for nodes which are not in AP's one-hop region, they can use the basic mode to communicate with their neighbors. In addition, for nodes not using CCMAC, they can still communicate with nodes using CCMAC, as long as they support the standard WiFi protocol.

Issues with packet size

As discussed earlier, packet shaping can increase the efficiency of concurrent transmissions. However, in some cases, packet shaping is not allowed, which mostly happens with some real-time application, e.g. V0IP. It may be too late to transmit if the VoIP application waits till the required size of data fills up. In fact, most of these applications specify a very small packet size. Using

RTS-CTS handshaking itself causes too much overhead for them. Hence, in CCMAC, if the packet size cannot be shaped, they should be sent directly to the AP with the legacy IEEE 802.11 protocol.

3.5 Learning of Coordination at AP

The good performance of CCMAC depends on the ability of the AP to perform proper coordination. However, it is a big challenge for the AP to learn this, because the network is stochastic and the AP does not have the full knowledge of the network, for example, the current channel condition between any TU to its helper. To solve this problem, we formulate it as a finite state Partially Observable Markov Decision Process (POMDP). This is because the nature of the AP coordination problem matches the model defined by POMDP, which is to make a sequence of decisions and maximize the average long term throughput (reward) in a stochastic environment, based on incomplete information.

3.5.1 Modelling the AP coordination problem as a POMDP

Suppose there are M nodes in the network. Out of these, there are N nodes which prefer to use two-hop relaying than direct transmission. The coordination problem is to consider these N nodes' information and make a sequence of decision to maximize data throughput. The mathematical model of POMDP has been introduced in chapter 2. From that we know, to model the coordination problem as a POMDP problem, we need to define the tuple of (S, A, P, R, O, Z) here:

State: The state contains the information of which nodes have already been selected to send packets and which nodes are holding packets to send, plus the information about the wireless channel. To represent it mathematically, $S = \{K_1, K_2, K_3, K_4, B_1, B_2 \dots B_N, Ch\}$. Here, K_1 always represents the main sender, K_2, K_3, K_4 represent the ID of the TUs which have already been selected. Since at most 5 nodes (see Theorem 3.1 in Section 3.6.1) can send packets simultaneously, there can be at most 4 nodes (1 sender and 3 TUs) that are selected before the last node. $B_i, i \in [1, N]$, is a binary number. $B_i = 0$ means node i currently has no packet to send, and $B_i = 1$ means node i currently has, at least one, packets to send. Lastly, Ch defines the channel characteristics.

Action ($a \in \{0, 1, \dots, N\}$): The decision is to put a certain node a into the list of TUs. $a = 0$ means choosing none of the nodes, and it is the end of the list. Note that, action selection can be executed up to 4 times within one transmission, because up to 4 nodes may be chosen as the TUs.

Observation: The coordinator, which is the AP, can observe a large portion of the state information including $K_1 \dots K_4$, $B_1 \dots B_N$. This is because, $K_1 \dots K_4$ are the ID of the nodes, which have already been selected by the AP. For $B_1 \dots B_N$, the AP assumes that nodes always have packets to send. Once a node has been selected as a TU, its corresponding B_j equals 0. The value will not change until the node j wins the contention. This means the node cannot be selected as a TU again until it clears the cached packet.

Reward $R(t)$: This equals the total throughput achieved, i.e. the aggregate throughput from the sender and TUs to the helper and HTUs, during

the coordination period t, when the AP receives the packet from the main sender. Otherwise, a large negative reward will be assigned. Note that, the AP can only know the throughput of the main sender immediately after the coordination. The throughput achieved by the TUs can only be known after those TUs win the contention. If the half mode is applied at that moment, the AP will know that additional throughput is achieved. Otherwise, the AP knows that the previous transmission failed. To calculate the rewards, we set a time limit and only take the node's throughput into consideration if the node wins the contention within the time limit.

The state transition model P and the state observation mapping model Z are dependent on the network topology and environment. These quantities are difficult to obtain. However, since we are adopting a model free Reinforcement Learning method to solve the POMDP, it is not necessary to know P and Z.

3.5.2 Using a RL algorithm to solve the AP coordination problem

Reinforcement Learning (RL) algorithms [38] can be used to solve MDPs and POMDPs. The goal of such algorithms is to find a policy that maps states or observations of the world to actions. Furthermore, RL algorithms focus on on-line performance, which involves finding a balance between exploration and exploitation. Hence, it can learn through interacting with the real world through a Monte Carlo-like method.

However, as discussed earlier, exact methods for solving POMDPs are

highly intractable. Hence, POMDP problems are often solved with approximation methods. One of them is to limit the set of policies to those representable by finite policy graphs, or finite-state controllers (FSC), of some bounded size. This allows us to achieve a compromise between the requirement that courses of action should depend on certain aspects of observable history, and the ability to control the complexity of the policy space. For the AP coordination problem, we employ the **IState-GPOMDP** algorithm proposed by Aberdeen [39], which uses a policy gradient approach with the FSC to approximately solve this problem. See **Algorithm 4** for the version of **IState-GPOMDP** algorithm adapted to the problem faced by CCMAC.

To use the FSC, we need to have an internal state I to represent the unobservable part of the real world. As proved in [40], if the size of the internal state approaches infinity, we can make the result as accurate as we want. With the FSC, the state is represented as the concatenation of the observation with the internal state, i.e. $S = \{O, I\}$. Hence, we can transform the POMDP problem to a MDP problem. For this problem, we make the size of the internal state equal five.

According to the IState-GPOMDP algorithm, we need to design two vectors ϕ and θ. ϕ is the parameter of the FSC and θ is the parameter of the policy. Parameterized by this two vectors, function $\omega(i_t|\phi_t, o_t, i_{t-1})$ determines the probability of choosing the internal state to be i_t at time t, and function $\mu(a_t|\theta_t, o_t, i_t)$ determines the probability of choosing the action a_t at time t. The two functions $\omega(.)$ and $\mu(.)$ have to be differentiable with respect to ϕ and θ.

Applying this algorithm to the AP coordination problem, we design two vectors θ^S and ϕ^S for each state S. Recall that $S = \{O, I\}$. The a^{th} element of vector θ^S represents the probability of choosing action a in that state. Similarly, the i^{th} element of vector ϕ^S represents the probability that the next internal state is i in the current state. Hence, θ and ϕ are not only the parameters of the functions $\omega(.)$ and $\mu(.)$, but can also represent the probability functions themselves. The gradient direction of $\mu(a|\theta, o, i)$ with respect to θ^S is:

$$\bigtriangledown \mu(a|\theta, o, i) = [\frac{\partial \theta_a^S}{\partial \theta_1^S}, \ldots, \frac{\partial \theta_a^S}{\partial \theta_{|A|}^S}] = [0, \ldots, 0, 1, 0, \ldots, 0] \tag{3.5}$$

Since θ_j^S is independent of θ_a^S except when $j = a$, the gradients are zero for all the elements except the a^{th} element, which equals 1. Similarly, the gradient direction of $\omega(i'|\phi, o, i)$ with respect to each element of ϕ^S, shown in equation 3.6 below, are all zero except the i' element, which equals 1.

$$\bigtriangledown \omega(i'|\phi, o, i) = [\frac{\partial \phi_{i'}^S}{\partial \phi_1^S}, \ldots, \frac{\partial \phi_{i'}^S}{\partial \phi_{|I|}^S}] = [0, \ldots, 0, 1, 0, \ldots, 0] \tag{3.6}$$

So far, we have defined all the terms required by the IState-GPOMDP algorithm. Now, we can use the policy gradient approach to solve the problem. The details of the algorithm is described in algorithm 4. A brief explanation of the algorithm is as follows: there is an infinite outer loop, which learns to coordinate forever. Inside the outer loop, there are three phases. The first phase is to act with the real world, i.e. the system will make a decision based on the current ϕ, θ and observation. In the meantime, it will record

Algorithm 4: Coordination using policy gradient algorithm

Given: The internal state I, internal state parameter $\phi(S)$, policy parameter $\theta(S)$ for each state; discount factor β and step size γ.

Set an arbitrary initial internal state i_0 and a random starting point of ϕ and θ, which fulfill the condition $\sum_{i=1}^{|I|} \phi_i^S = 1$ and $\sum_{a=1}^{|A|} \theta_a^S = 1$, for all states.
while *System running* **do**
  Set $t = 0$, $k = 1$
  while $t < T$ **do**
    Get the observation o_t from the world.
    Choose the internal state i_{t+1} based on $\phi(o_t, i_t)$.
    Choose the action a_t based on $\theta(o_t, i_{t+1})$ and put this action into the k^{th} position of the list.
    if $a_t == 0$ *or* $k == 4$ **then**
      Send the list, set $k = 1$
    else
      $k = k + 1$
    end
    Get the reward r_t.
    Record these o_t, i_{t+1}, a_t, r_t for the learning process in the next loop.
    $t \leftarrow t + 1$
  end
  Set $t = 0$
  Set vectors $z_0^\phi(S) = [0]$, $z_0^\theta(S) = [0]$, $g_0^\phi(S) = [0]$, $g_0^\theta(S) = [0]$ for all states. Here $z_0^\phi(S), g_0^\phi(S) \in \Re^{n_{\phi(S)}}$, $z_0^\theta(S), g_0^\theta(S) \in \Re^{n_{\theta(S)}}$
  while $t < T$ **do**
    $z_{t+1}^\phi(S) = \beta z_t^\phi(S) + \frac{\nabla \omega(i_{t+1}|\phi, o_t, i_t)}{\omega(i_{t+1}|\phi, o_t, i_t)}$
    $z_{t+1}^\theta(S) = \beta z_t^\theta(S) + \frac{\nabla \mu(a_t|\theta, o_t, i_{t+1})}{\mu(a_t|\theta, o_t, i_{t+1})}$
    $g_{t+1}^\phi(S) = g_t^\phi(S) + \frac{1}{t+1}\left[r(t+1) z_{t+1}^\phi(S) - g_t^\phi(S)\right]$
    $g_{t+1}^\theta(S) = g_t^\theta(S) + \frac{1}{t+1}\left[r(t+1) z_{t+1}^\theta(S) - g_t^\theta(S)\right]$
    $t \leftarrow t + 1$
  end
  forall *states of* S **do**
    $\phi(S) \leftarrow \phi(S) + \gamma g_T^\phi(S)$
    $\theta(S) \leftarrow \theta(S) + \gamma g_T^\theta(S)$
    For the elements of $\phi(S)$ and $\theta(S)$, which is negative, set the value equals 0
    $\phi(S) \leftarrow \frac{\phi(S)}{\sum_{i=1}^{|I|} \phi_i(S)}$; $\theta(S) \leftarrow \frac{\theta(S)}{\sum_{a=1}^{|A|} \theta_a(S)}$
  end
end

down the necessary information for the learning phase. The second phase is to find the gradient direction of the ϕ and θ by learning. It uses the records from the first phase to estimate the gradient, which is represented as g. The last phase is to update the values of ϕ and θ, to which the estimated gradient g is added, before the values are normalized to fulfill the probability constraint.

3.6 Analysis

3.6.1 The maximum number of concurrent transmissions

Theorem 3.1 *We assume that all nodes have the same circular transmission region with radius equal to the maximum transmission range r. The interference region for each node is the same as the transmission region. Secondly, we assume that nodes which are closer to each other will have a higher transmission rate. Then, there is a maximum of 5 relay processes which can co-exist, without interfering each other, in the one-hop transmission area of the AP.*

Proof: We define the one-hop transmission region of the AP as the unit circle. Referring to Figure 3.3 that, node O represents the AP. We observe that, the transmission or interference area of any node (e.g. node P in Figure 3.3) in the unit circle equals to the intersection area of two circles with same radius r. In Figure 3.3, it is shown as the area $A \bigcup B$. This area can be divided into two parts. One is the relay area, which is shown as the area B (the shaded area) in Figure 3.3. The other is the non-relay

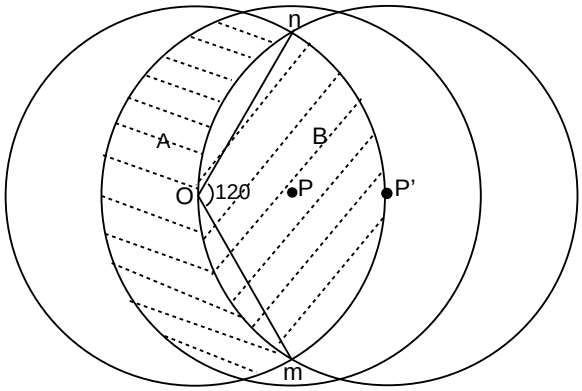


Figure 3.3: The intersection area and the relay area.

area, which is shown as the area A. The relay area, B, is defined as the intersection area when the sender is located at the edge of the unit circle. For any sender, we can always find such an area by projecting its location to the edge. An example of such a projection, from point P to P', is shown in Figure 3.3. It can be easily proved that the size of the relay area for any sender is the same; and the angle for $\angle mOn$ is always 120°. In addition, because, the sender's location is on the line O, P', it is easy to prove that if there are any helper nodes located at the non-relay area, the distance from the sender to the helper will be larger than the distance from sender to the AP directly. This means that, the rate for two-hop transmission is lower than that of direct transmission. Hence, helper nodes should not be chosen in the non-relay area, A, but only in area B, the relay area.

We can also observe that, within the relay area, there is a circular sector m, O, n with size $\frac{1}{3}\pi r^2$. Since each sender covers its relay area, it also covers this circular sector correspondingly. One important feature of these circular sectors is that, if a point is covered by 3 or more such circular sectors, for

the sectors in the middle, their corresponding relay areas will be completely covered by the union of the two relay areas at the left most and the right most. To prove this, we can draw a line from the center of the unit circle, O, to the point Q, which is covered by 3 sectors. For the circular sector in the middle, the corresponding relay area at the left side of the line (O, Q) is completely covered by the left most relay area, and the area at the left side of the line is covered by the right most relay area. This important feature tells us that if there are multiple senders doing the two-hop relaying simultaneously, there should be no point in the unit circle, covered by 3 or more circular sectors, which corresponds to different senders. Otherwise, some of the senders cannot find any helpers in the relay area which will not be interrupted by the two nodes at the side.

The rest of the proof is simple. Since we know that the size of the circular sector is $\frac{1}{3}\pi r^2$ and any point in the unit circle can be covered by at most 2 circular sectors. There are at most 6 circular sectors which can co-exist. However, at that time, every point in the circle is covered by 2 sectors, which means that there are interruptions everywhere. Hence, 6 concurrent transmissions are not feasible. Finally, we find that 5 is a feasible value. One example of such a topology is shown in Figure 3.4, where the capital letters represent the senders and smaller letter represent the helpers. It is clear that, each helper node can receive the message from its corresponding sender and cannot receive message from any other senders.

∎

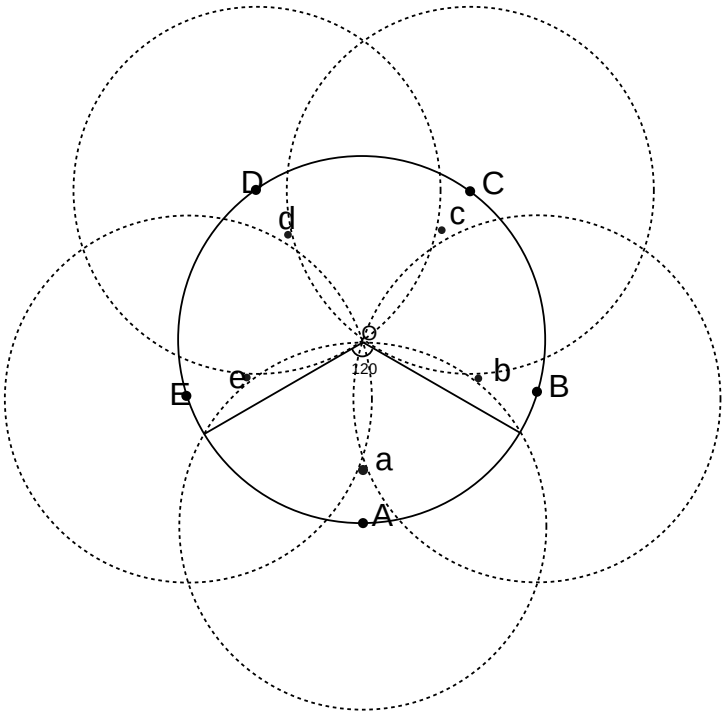


Figure 3.4: The intersection area and the relay area.

3.6.2 The average transmission time to send a packet

Suppose, each time the AP specifies m TUs on average, the bit error rate is p_f^b. Hence, the loss rate for a packet is $p_f^P = 1-(1-p_f^b)^n$, where n represents the packet length. Since the control packets, including RTS, CTS, HTS and ACK, have small sizes; and in addition, they are sent by the lowest rate, the packet loss rate for them are very small. For simplicity, we assume the loss rate for a control packet is zero. We also assume the extra delay caused by the hardware circuit, physical layer preamble is zero. For nodes using the CCMAC protocol, the average transmission time for a node transmitting

one packet is:

$$T_{cc}^{overall} = \frac{p_e}{p_e + p_h} T_{cc}^{en} + \frac{p_h}{p_e + p_h} T_{cc}^{half} \tag{3.7}$$

p_e and p_h represent the probability of transmitting using the enhanced mode or the half mode respectively. The ratio between p_h and p_e is: $p_h = m(1 - p_f^P)p_e$. T_{cc}^{en} and T_{cc}^{half} is the average transmission time for the enhanced mode and half mode in CCMAC, respectively.

$$T_{cc}^{en} = \frac{1-(1-p_f^P)^2}{(1-p_f^P)^2} T_{fail}^{en} + T_{succ}^{en} \tag{3.8}$$

$$T_{cc}^{half} = \frac{p_f^P}{1-p_f^P} T_{fail}^{half} + T_{succ}^{half} \tag{3.9}$$

$$\begin{aligned} T_{fail}^{en} = {} & T_{RTS} + T_{HTS} + T_{CTS} + \frac{L}{R_{s,h}} + \frac{L}{R_{h,d}} \\ & + T_{DIFS} + 4T_{SIFS} \end{aligned} \tag{3.10}$$

$$\begin{aligned} T_{fail}^{half} = {} & T_{RTS} + T_{HTS} + T_{CTS} + \frac{L}{R_{h,d}} \\ & + T_{DIFS} + 3T_{SIFS} \end{aligned} \tag{3.11}$$

$$T_{succ}^{en} = T_{fail}^{en} + T_{ACK} + T_{SIFS} \tag{3.12}$$

$$T_{succ}^{half} = T_{fail}^{half} + T_{ACK} + T_{SIFS} \tag{3.13}$$

T_{fail}^{en} and T_{succ}^{en} is the transmission time for a failed attempt using enhanced mode or a successful attempt using enhanced mode, respectively. Similarly, T_{fail}^{half} and T_{succ}^{half} is the time spent for a failed attempt or a successful attempt, using half mode, respectively. Noted that the contention time, which is related with the number of contention nodes in the network, has not been

included.

3.7 Simulations and Results

3.7.1 Simulation Setup

The simulation is done using OMNet++ [41] to simulate an IEEE 802.11b network. For this network, an AP is at the center of a unit circle with radius equal to 100 m. Two types of nodes are scattered in two different areas. Relay nodes are uniformly and independently distributed in the inner circle, with a radius of 67.1 m. Senders are uniformly and independently distributed in the outer ring, with inner radius equal to 67.1 m and outer radius equal to 100 m. By this setting, the direct transmission rate between the sender nodes and the AP is less than 2 Mbps. The relay nodes do not generate any data, while the sender nodes generate data and send them to the AP. In addition, since we concentrate on the throughput performance, sender nodes are always backlogged. Following [5], other simulation settings are as follows: The size of RTS packet, CTS packet and HTS packet are 352 bits, 304 bits and 304 bits, respectively. DIFS is set as 50 μs, SIFS is set as 20 μs. The effective communication range for data rate of 11 Mbps, 5.5 Mbps, 2 Mbps and 1 Mbps are 48.2 m, 67.1 m, 74.7 m and 100 m, respectively.

The channel loss is modeled by two factors which are the bit error rate (BER), and the channel fading process. The BER is set as 1×10^{-4}. The fading process is modeled as a two state Markov chain [42] [43], either the link is "up" or "down". The probability of a link going down is p_1, and the

probability of a link going up is p_2. In this simulation, we let $p_1 = 0.05$, $p_2 = 0.2$ and the duration for each state is a uniformly distributed random number between $0 - 2$ seconds.

We study the performance of five different protocols: WiFi, CoopMAC, CCMAC, random$_1$ and random$_4$. WiFi is the legacy 802.11b DCF protocol with channel rate adaptation, i.e. the RBAR extension. CoopMAC is the protocol described in [5], and is the main comparison target for CCMAC. CCMAC is the protocol proposed in this paper. Note that, before each experiment, expect experiment 1, we let the AP learn how to coordinate for 600 seconds before taking the results. The protocols of random$_1$ and random$_4$ have the same cooperation part as CCMAC. However, in these protocols AP performs random coordination, i.e. no learning. Random$_1$ chooses one TU randomly and random$_4$ chooses four TUs randomly. Hence, they have a much higher probability that the cached data (the data of the TUs) encounter transmission failure. However, for the main sender, i.e. the node which wins the contention, its packet transmission is still protected by the HTS. We use these two protocols to compare the performance difference between non-coordinated protocols and CCMAC.

3.7.2 Experiments

In the first experiment, there are 8 sender nodes randomly scattered in the outer ring area, 20 relay nodes randomly scattered in the inner circle area and the reference packet size is 1024 bytes. We are interested to see the learning process of the RL algorithm implemented in CCMAC. The results

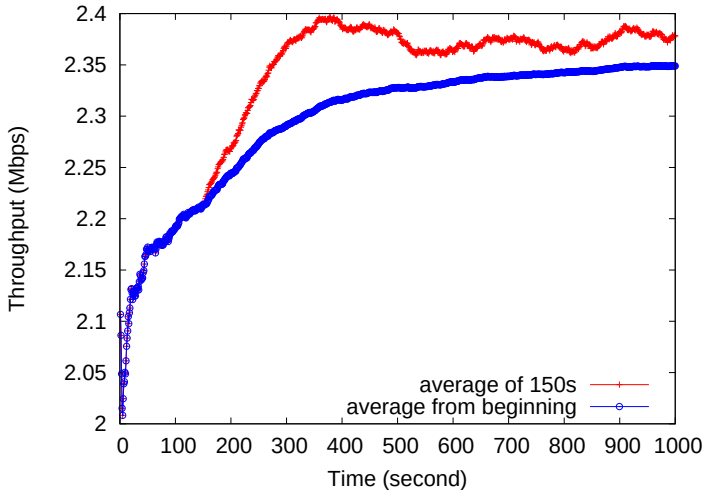


Figure 3.5: The average throughput achieved while learning.

are shown in Figure 3.5. The lower line shows the average throughput performance starting from time 0. The upper line shows the average throughput performance of the past 150 seconds. We can see that, the learning process takes around 300 seconds to reach the long time average point, 2.36 Mbps. Then, the average performance fluctuates with the dynamics of the networks. This experiment illustrates the learning speed of the RL algorithm and the throughput performance during the learning process.

In the second experiment, there are 8 senders randomly scattered in the outer ring area, and the reference packet size is 1024 bytes. We vary the number of relay nodes to see the performance of different protocols. The average results of 10 random topologies are shown in Figure 3.6. Clearly, the performance of WiFi is not affected by the number of relay nodes, since it does not use relaying. Hence, it gives a flat line. All the other protocols benefit when there are more relay nodes. As expected, among them, CC-MAC reaps the most benefit and it outperforms CoopMAC by around 15%;

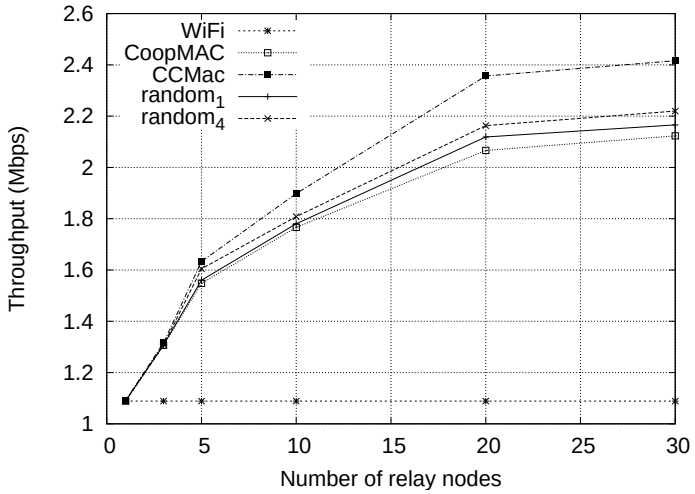


Figure 3.6: The average throughput achieved with different numbers of relay nodes.

and it outperforms the random_1 and random_4 around 10%.

In the third experiment, we set the number of the relay nodes to 20 and the reference packet size as 1024 bytes. We randomly add in more sender nodes to see the performance difference. We repeat these experiments for 10 different topologies. The average results are shown in Figure 3.7. We also randomly pick 3 sets of result from these 10 individual experiments, which are shown in Figure 3.8(a), Figure 3.8(b), and Figure 3.8(c). From the graphs, although the performance of all the protocols is affected by the senders' random location, everytime the performance of CCMAC is still the best among all the four protocols. As the right part of the Figure 3.7 shows, with more senders, the performance of CoopMAC gradually decreases due to more contention. However, CCMAC gives a different trend: with more senders, it may even achieve better performance. The reason is that, with more senders, it may be able to choose more nodes to transmit simultane-

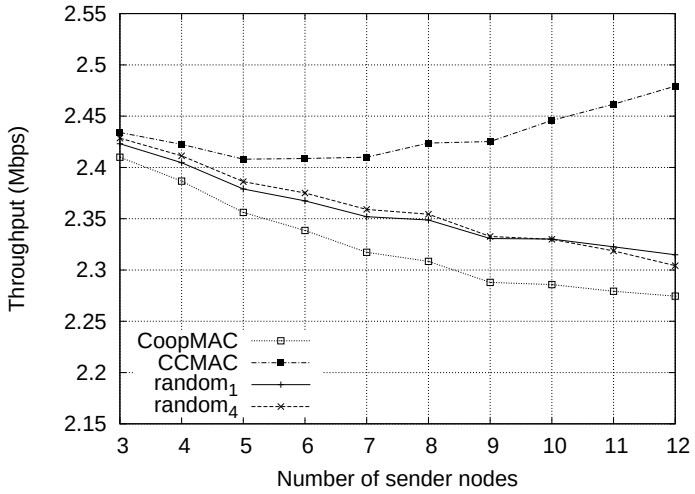
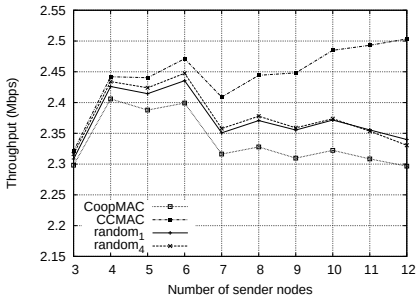
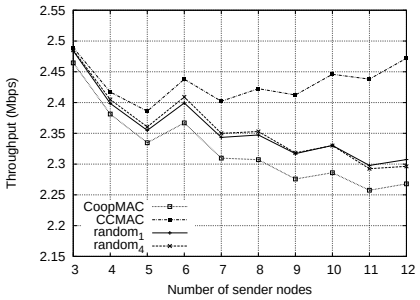


Figure 3.7: The average throughput of 10 topologies achieved with different numbers of sender nodes.

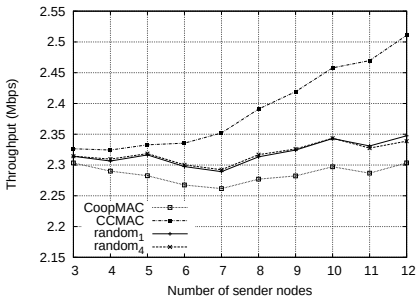
ously. It shows that, when the number of nodes is increased, protocols with concurrent transmissions capability, such as CCMAC, reap more benefits. In the last experiment, we set the number of relay nodes to 15 and the number of senders as 8. We choose different reference packet sizes to see the performance of different protocols. The results are shown in Figure 3.9. Clearly, packet size is one of the most important factors which affects the throughput. However, the effect is different for different protocols. We can see from the results that, compared to CCMAC and CoopMAC, the change of packet size has the less impact on the WiFi protocol. The reason is that, CoopMAC and CCMAC have larger overhead than WiFi since WiFi does not have the HTS packet. When the packet size is small, the throughput gain from the two-hop transmission is canceled by the extra overhead. However, when the packet size increases, the gain from two-hop transmission increases, leading to an increase in throughput. Another result shown in the graph is that the



(a) Case One.



(b) Case Two.



(c) Case Three.

Figure 3.8: The throughput performance based on 3 different network topologies.

change of packet size has a larger effect on CCMAC than CoopMAC. The reason can be shown mathematically. Suppose the transmission times of sending a packet by CCMAC and CoopMAC are T_{cc} and T_{coop}, respectively. Both of them have two components. Firstly, the overhead time, C, which includes the time to transmit the control packets, and the time of waiting. Clearly, they are not related to packet length and the C for both protocols are almost similar, Hence, we have $C_{cc} = C_{coop}$. The second component is the transmission time of payload, and obviously it is related to the packet size. We define the coefficient k and use $k \times L$ to represent this. Since, CCMAC takes less time than CoopMAC in transmitting a fixed amount of data, $k_{cc} < k_{coop}$. Hence, the ratio of the throughput between CCMAC and CoopMAC can be represented as $\frac{Thr_{CC}}{Thr_{coop}} = \frac{C_{coop}+k_{coop}L}{C_{cc}+k_{cc}L}$. Clearly, when L increases, the ratio increases. This means that, when packet size increases, the throughput of CCMAC increases faster than CoopMAC and vice versa.

3.8 Conclusion

In this chapter, we explored the benefits of cooperative communication and concurrent transmissions at the medium access control (MAC) layer in the AP's one-hop region. We proposed a novel coordinated cooperative MAC (CCMAC) protocol which utilizes these features to improve the throughput performance of the network. CCMAC takes three steps before transmission: rate detection, helper selection and packet shaping. CCMAC has three different transmission modes: basic mode, enhanced mode and half mode. One

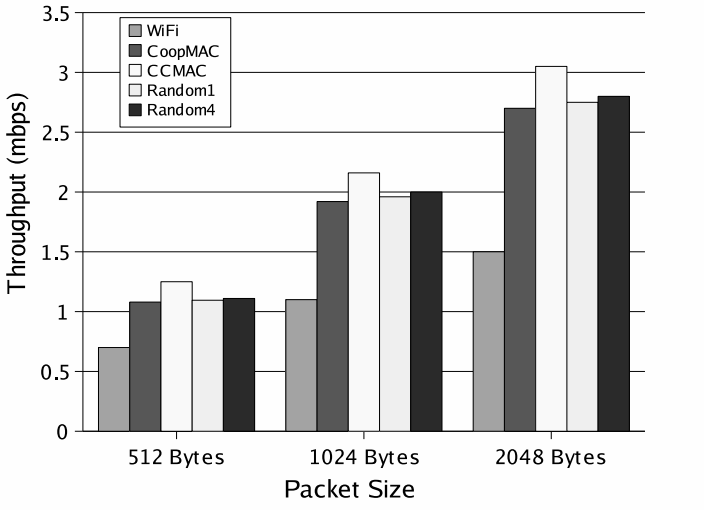


Figure 3.9: Average throughput achieved with different packet size.

of the modes is chosen based on the channel condition and the helper's status. The enhanced mode enables up to 5 concurrent transmissions, which requires good coordination between nodes. CCMAC achieves this by treating the coordination problem as a POMDP and using a policy gradient algorithm based on reinforcement learning to solve it. Through analysis and simulation, we verified that CCMAC can achieve substantial throughput performance improvement, without incurring significant network overheads. We have also argued that even if all nodes are selfish, it is still in their interest to cooperate, since by using CCMAC, all nodes, including the helpers, benefit from such cooperation and have almost nothing to lose.

Chapter 4

Concurrent Cooperative MAC (downlink)

After studying the MAC protocol of the uplink in wireless LAN, in this chapter, a MAC protocol of the downlink is introduced. That is the SI-CCMAC (Sender initiated CCMAC). Similar as CCMAC, SI-CCMAC applies the techniques of opportunistic cooperation and concurrent transmission. The main difference between them is at the transmission procedure and the coordination scheme between different nodes. Hence, in this chapter, we will not introduce the common issues such as the motivations and related works. Instead, we will concentrate on the cooperation and coordination mechanism of SI-CCMAC, together with its performance in network.

4.1 The Transmission process

SI-CCMAC has the similar process of the transmission rate detection, helper selection and packet shaping as the uplink. Hence, we will not introduce these techniques again. In SI-CCMAC, before each transmission, a destination list is generated at the back-end, which is to be discussed in detail in the next section. This list indicates one or multiple destination nodes, which are to receive their packets in the current transmission. Based on this destination list and node's helper tables, different transmission mode will be selected. In SI-CCMAC, there are three different transmission modes. The first is direct mode, the second is two-hop mode and the last is multi-destination mode. To better illustrate the transmission process for two-hop mode and multi-destination mode, diagrams of data flow sequence are shown based on a sample network scenario, of which the topology is shown in Figure 3.1.

The direct mode happens when there is only one node in the destination list and this node has no helper. This time, direct transmission (same as WiFi) will be used. RTS/CTS messages are exchanged before the data packet's transmission, and the data packet is sent directly to the AP.

The two-hop mode happens when there is only one node in the destination list and this node has at least one helper. This time, the rDCF/CoopMAC kind of transmission will be used. RTS (request from sender), HTS (confirmation from help), CTS (confirmation from destination) messages are exchanged before the data packet's transmission, and the data packet is sent by the 2-hop relaying. Figure 4.1 shows the data flow sequence for the

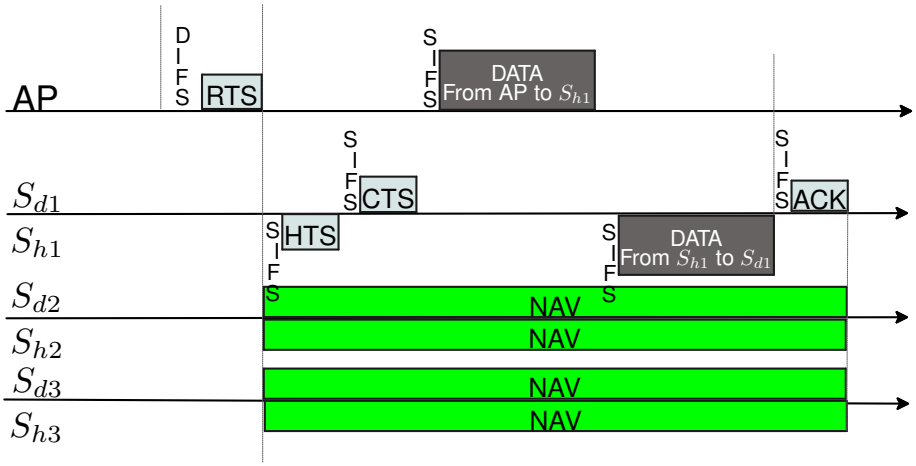


Figure 4.1: Example: message flow for two-hop mode

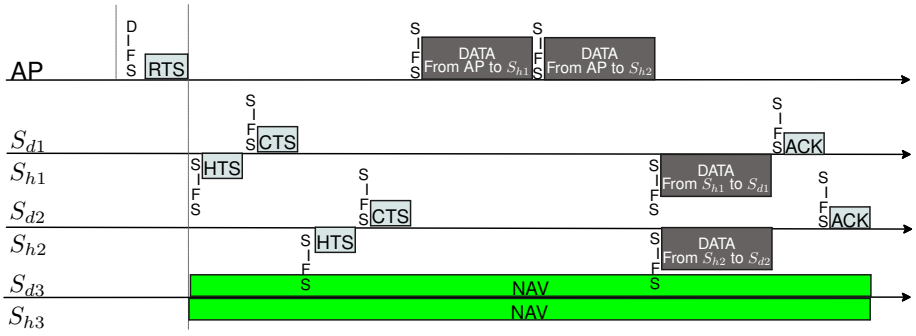


Figure 4.2: Example: message flow for multi-destination mode.

two-hop mode transmission. In this example, AP sends packet to node S_{d1} through relaying.

There is a special case in two-hop mode, that is, when the CTS is received by the sender, but not the HTS message. It means, this time, the AP assumes that the helper is not ready to help, but the destination node is ready to receive the packet. Hence, instead of requesting the channel again, the data is sent directly to the destination node with the direct channel rate.

The multi-destination mode happens when the destination list contains multiple nodes. The procedure is described as follows. Firstly, the AP sends out the RTS message, which includes the IDs of all the selected destination nodes and their correspondent helpers. Then, each of the helper-destination pair to HTS followed by CTS messages according to the sequence given by RTS message. If both the HTS and CTS messages, from one helper-destination pair, are received by the AP, AP will put this pair into the To-delivery (TD) list. After all pairs sending back their response, AP will send the corresponding data packet to the helpers, in the TD list, one after another. After all helpers, in the TD list, having received the corresponding data packets from AP, they will forward the data packet to the corresponding destination nodes simultaneously. Finally, the destination nodes sends the ACK messages to the AP one after another. Figure 4.2 shows the data flow sequence for the multi-destination mode transmission. In this example, AP sends packets to both node S_{d1} and S_{d2} through relaying.

4.2 SI-CCMAC back-end: Downlink Allocation

The good performance of SI-CCMAC depends on the ability of the AP to create a good destination list before each transmission. Equivalently, it is to allocate the downlink to one or multiple nodes at different moment. There are two objectives of the allocation. The first objective is to ensure a max-min fairness among different nodes. The second one is to optimize the overall throughput performance, which is equivalent to maximize the average data rate. In this section, we try to separate these two goals by tackling the issue of fairness first. After relaxing the constraint of fairness, the optimization problem is formulated as a vertex coloring problem, a maximum independent set problem or a MDP problem, under different assumptions, and can be solved accordingly.

4.2.1 Solving the fairness constraint

Fairness is an important performance metric for network performance. In many cases, optimizing throughput performance without fairness guarantee is worthless. Since the "best" scheme might optimize network throughput while denying access to a particular (or a set of) user(s).

Weighted Fair Queuing (WFQ) [44] technique is widely preferred to achieve Max-min fairness in channel allocation. In WFQ, each packet is assigned a sequence number when they arrive, as described in [44]. Each time the packet with the smallest sequence number will be selected as the header packet (HP) and to be served (transmitted). However, WFQ is designed for sequential packet delivery. To handle concurrent transmission,

a simple extension of WFQ, which is called C-WFQ is proposed. The C-WFQ has the same concept of weight as WFQ and it implements the same sequence number assignment procedure as WFQ. However, besides selecting HP, the C-WFQ creates a List of Packets-to-be-Considered (LPC). This LPC contains all the packets with the sequence number Q: $Q < Q_{HP} + \tau$, where Q_{HP} is the sequence number of the current header packet and τ is a pre-set constant with a positive real value, which regulates the tolerance gap. Clearly, if $\tau = 0$, C-WFQ is identical to WFQ, because no packets can be selected in LPC. It can be easily seen that, if only packets, listed in LPC, are selected to take the concurrent transmission with HP, max-min fairness can be achieved in the long run. In addition, similar as WFQ, the packet delay is also bounded if the incoming data flow is leaky bucket constrained.

4.2.2 A simplified problem

If a network satisfies all the following 3 assumptions, we can simplify the coordination problem and model it as a vertex coloring problem. The first assumption is that: the network topology is stable. It means, nodes are stationary and no nodes leave or enter the network. Secondly, the noise level for all the channel is zero. This means the packet delivery is always successful unless a collision happens. Lastly, all nodes always have packets to receive from the AP. Although these assumptions are non-practical, only under this case, we may get the long term optimal solution. Because, these assumptions help to remove all the uncertainty of the future.

With the last assumption, the fairness constraint can be relaxed if each

node receives a same amount of data in each round. With the presence of rounds, the objective of maximizing the average throughput performance for infinite steps now becomes to optimize the average data rate for each round. Clearly, it is equivalent to minimizing the total transmission time (the time for one round) that a fixed amount of data is transmitted to every node. As shown in Theorem 4.1 that this is also equivalent to minimizing the number of 2-hop transmissions, required for one round. The term 2-hop transmissions includes the case of one destination transmission and the case of multiple destination transmissions, i.e. if a concurrent 2-hop transmission sends different packets to n destination at same time, it is still regarded as one 2-hop transmission.

Theorem 4.1 *Under the assumptions that 1) the network topology is stable; 2) there is no noise in the wireless medium; 3)nodes always have packets to receive; to minimize the total transmission time that every node receives a fixed amount of data, is equivalent to minimizing the number of 2-hop transmissions required for one round.*

Proof: We would like to prove for the case, where at most 2 packets can be transmitted concurrently. Because this is easier to be understood and the process of this proof can be easily adapted to the cases where more than 2 packets are allowed to be transmitted together.

Firstly, we would like to formulate the problem of minimize the total

transmission time as an integer optimization problem, which is shown below:

$$
\begin{aligned}
\min \quad & T_{total} \\
\text{s.t.} \quad & T_{total} = \sum_{n=1}^{N} k_n^{di} T_n^{di} + \sum_{m=1}^{M} \sum_{j=m}^{M} \beta_{m,j} T_{m,j}^{re} \\
\forall p: \quad & p \in \{1, 2, \ldots, M\}, \sum_{m=1}^{m<p} \beta_{m,p} + \sum_{j=p}^{M} \beta_{p,j} = k_p^{re} \\
\forall m, j \quad & m, j \in \{1, 2, \ldots, M\} \ \& \ (m < j), \\
& \qquad T_{m,j} = T_{sh}(m) + T_{sh}(j) + T_{hd} \\
& \beta \in \{1, 2, 3, \cdots\}
\end{aligned}
\tag{4.1}
$$

where N is the number of nodes preferring direct transmission and M is the number of nodes preferring relay-based (2-hop) transmissions. The variable k, which is a set of given values, represents the number of packets to be received by any node within one round. For example, k_n^{di} and k_m^{re} are the number of packet to be received within one round by node n and m, given node n using direct transmission and node m using two-hop transmission. T_n^{di} is the transmission time of node $n's$ one packet, where direct transmission is applied. $T_{m,j}^{re}$ is the transmission time of a concurrent transmission during which node m and node j both receive one packet. When $m = j$, $T_{m,j}^{re}$ is the transmission time of a non-concurrent 2-hop transmission, during which node m receives one packet. $\beta_{m,j}$ is the number of transmissions, in a round, that node m and $j's$ packets are sent at the same time. For example, if $\beta_{1,2} = 3$, it means the scheduler will assign slots, for node 1 and 2 to take the concurrent transmission, 3 times in a round. Hence, $\beta_{m,j}$ forms a 2-dimensional matrix which determines the actual strategy. [1]

[1] If more than 2 nodes are allowed to send concurrently, the β grows with higher dimension.

Since, the packet size is shaped, the time of the transmission from the helper to the destination is the same for all the helper-destination pairs, i.e. the T_{hd} is a constant for all 2-hop transmissions. We can substitute equation 4.1 and equation 4.1 into equation 4.1 and have the following equation:

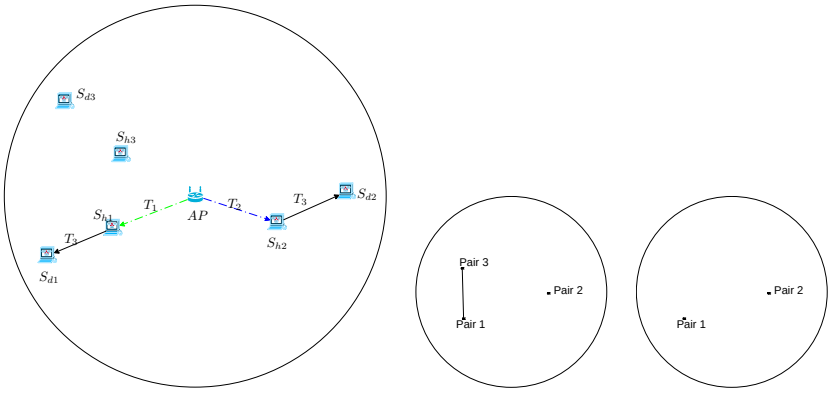
$$T_{total} = \sum_{n=1}^{N} k_n^{di} T_n^{di} + \sum_{m=1}^{M} k_m^{re} T_{sh}(m) + \sum_{m=1}^{M} \sum_{j=m}^{M} \beta_{m,j} T_{hd} \tag{4.2}$$

Clearly, all the k_n^{di}, T_n^{di}, k_m^{ro}, $T_{sh}(m)$ and T_{hd} are fixed values. Hence, T_{total} is only related with the value of $\sum_{m=1}^{M} \sum_{j=m}^{M} \beta_{m,j}$, which is the total number of 2-hop transmissions required for one round. Minimizing the total transmission time is equivalent to minimizing the total number of 2-hop transmissions. ∎

Creating the conflict graph

With the assumption that nodes are stationary, we can model the network as a stationary graph. In addition, because of Theorem 4.1, we only need to concentrate on the scheduling of the nodes with 2-hop transmissions. Hence, we create a *conflict graph* $G_c = (V, E)$ (similar as [45]) based on the stationary network topology, where V is the set of vertics and E is a set of edges. Suppose there are N helper-destination pairs in the network[2]. Then $|V| = N$, and each vertex (denoted by v_i) of G_c corresponds to the i^{th} helper-destination pair. There is an edge between vertex v_i and vertex v_j in G_c, if either the i^{th} pair or the j^{th} pair conflicts with the other due to interference. Hence, any two transmissions (two vertics), which are connected

[2]Some of the destination nodes may share the same helpers. However, they are still considered as different pairs.



(a) The network topology and the flow of messages.

(b) The related conflict graph.

(c) The deduced temporary graph.

Figure 4.3: A sample network.

in the conflict graph cannot transmit at same time. In real implementation, each destination node needs to report to AP its conflict node list. It includes all the nodes (except its own helper), whose received signal power is too high compared with the helper to be tolerated. Then AP draws the edge based on the conflict node list. It means that, there will be an edge between v_i and v_j, if the destination node of pair i reports to AP that it cannot tolerate the helper of pair j, or the destination node of pair j reports to AP it cannot tolerate the helper of pair i. Figure 4.3(b) shows the conflict graph based on the network topology in Figure 4.3(a).

Modelling as a weighted vertex coloring problem

With Theorem 4.1, we can model the optimization problem as a weighted vertex coloring problem. A weighted vertex coloring problem is an extension

of vertex coloring problem, which is to find a way of coloring the vertexes of a graph such that no two adjacent vertexes share the same color. And the goal is to find a way which uses the minimum number of colors. To model the coordination problem as a vertex coloring problem, we need to create the *conflict graph*, the steps of which are shown in the last section.

Once, the conflict graph is created, to schedule the concurrent 2-hop transmissions is equivalent to coloring the conflict graph. We can assign the same color to different vertex as long as they are not connected. Equivalently it means, we can let these two pairs of nodes to transmit at the same time without collision. Hence, to have the minimum number of 2-hop transmissions (including concurrent 2-hop transmissions) is equivalent to using the minimum number of colors to color all the vertexes, which is indeed a vertex coloring problem. However, unlike the normal vertex coloring problem that each vertex just needs to be colored once, in this cases, each pair may send different numbers of packets within one round. Hence, the problem have to be modeled as a weight vertex coloring problem, which has been introduced in Chapter 2.2.1. In this case, the weight of each vertex is the number of packets required to be sent in each round. As introduced before, the complexity of this problem is proved to be NP-complete [46]. However, if the network is small, which is normally true in the single hop networks, we can compute the exact optimal solution using integer programming, which is shown in the proof of theorem 4.1. If the network is large, we can approximately solve this problem using any proposed algorithms for the weighted vertex coloring problem, such as [47]. The performance of these algorithms

are bounded within a factor of the optimal solution and their computation complexities are polynomial.

4.2.3 The general case

Although we can get the long term optimal solution using the model of weighted vertex coloring, this model cannot be applied to a generic network. It means that, the constraint optimization problem still has not been solved for the general cases, where the packet incoming rate and the wireless channel behavior are unpredictable. We know that the action taken at the present time not only affects the immediate performance but also influences the future performance. Since the future is unpredictable, at the present time, there is no way to know the part of influencing. Hence, the long term optimization is not achievable. The way we proposed is to find the greedy algorithm, which optimizes the one step immediate performance.

As discussed before, the fairness constraint in general case, can be relaxed by C-WFQ algorithms, where packets, with sequence numbers less than $Q_{HP} + \tau$ can be sent together with the header packet (HP). After relaxing the fairness constraint, there are two cases to be considered. Either the HP (header packet) prefers direct transmission or the HP prefers two-hop transmission. In the first case, there is nothing we can do but send the HP directly. In the second case, nodes in the LPC can be selected. Hence, the goal of the greedy algorithm is to find the strategy that maximizes the number of nodes taking concurrent transmissions without packet collision. However, before we introduce the solution, we want to create a more special

graph *temporary graph* based on the idea of *conflict graph*.

Creating the temporary graph

The *conflict graph* is stationary as long as the network topology is stationary. However, it lacks other information such as whether there are packets for any destination node. Such information is changing dynamically. Hence, we need to create a sub-graph of the *conflict graph* including all the possible pairs which can be selected and excluding all the pairs which cannot be selected. There are three steps to construct this temporary graph $G(t)$ based on the *conflict graph* G_c. Firstly, in order to achieve the fairness constraint, delete the vertexes, which do not have packets contained in LPC. Secondly, in order to ensure that the HP is selected, all the vertexes, which have an edge between the HP and themselves need to be deleted. Finally, we delete all the edges in the graph, which connect with these deleted vertexes. Figure 4.3(c) shows an example of getting the $G(t)$ based on G_c. In this example, we assume that all pairs have packets and these packets' sequence numbers all satisfy the requirement given by C-WFQ (introduced in Section 4.2.1. Among them, a packet for node S_{d1}, i.e. pair 1, is the current HP.

Our goal is to schedule the maximum number of pairs doing concurrent transmissions with the pair corresponding with HP, while ensuring collision free. It is equivalent to select the maximum number of vertex on the $G(t)$ while ensuring that all the selected vertexes are not directly connected, which is to find the maximum independent set [48] in the graph $G(t)$. The complexity of finding the optimal solution of MIS problem is NP-hard. However, if the number of vertex in the graph is small, we can still try to find

the optimal solution by Integer Programming. Otherwise, algorithms like [28] can be used to approximately solve this problem.

4.2.4 MDP Modelling

In the general case, we assume that the future is unpredictable, hence it is impossible to find the optimal solution. However, like some cases in real life, the near future may be predictable based on the current situation. Hence, the questions are: in this problem, can we have a stochastic prediction of the future? Furthermore, how to improve the performance of the network based on the prediction?

To formulate this situation, we assume that nodes are stationary. We also assume that the packet arrival rate at the AP follows a Poisson distribution for each data flow and the AP stores all incoming packets in different queues based on different destination and each queue has a length of L_{max}. Finally, we assume that the wireless channel has a constant packet loss probability [3]. With these assumptions, we would like to model this problem as a A finite state MDP.

Modelling the AP coordination problem as a MDP

As shown in [49], in a wireless network with N nodes, the queue length of each node can be viewed as the following Markov process: $L_i(t+1) = (L_i(t) + arr_i(t) - u_i(t))^+$. Here, $L_i(t)$ represents the queue length of node i at time t; $arr_i(t)$ is the number of bits arriving to the queue; $u_i(t)$ is

[3] With a time varying channel, it can be modeled as a partially observable MDP, which is more complex

the number of bits transmitting out of the node and $(L)^+ := max(L, 0)$. The above process can be mapped to the AP scheduling problem. This time, the n nodes can be viewed as n data flows. Based on the assumption, the arrival process for each flow is still Poisson. However, the transmission process for each individual flow is controlled by the action chosen by the AP. Once the AP sends data to one or a few nodes, the transmitting process for the corresponding nodes are non zero and can be estimated, while being zero for all the other flows. After aggregating all the n data flows, these individual Markov processes form a big controlled Markov process with n different queue length as the state space and the next state is stochastically controlled by AP's action. On the other hand, AP can just choose the action based on the queue length information. It means, the action chosen at each step can be determined by a policy π: with $a(t) = \pi(L_1(t), L_2(t), \ldots L_N(t))$. Then, the queue length (states) of the next time step will be a function of this action and the current queue length. However, because we have a new fairness constraint, which is handled by C-WFQ. The action chosen at each step does not fully depend on the queue length information. It is also affected by the current HP and the sequence number of the first packet in each flow. Hence, they have to be considered in the state space. Furthermore, the transition (next step value) of the sequence number of the first packet depends on the sequence number of the second packet, ect. In the end, in order to show that it is Markovian, or in other words to show the dynamics of the state transitions, the sequence numbers of all the packets have to be taken into consideration.

Now we can define the complete MDP formulation. Suppose there are n nodes in the network. The AP has provided a queue for each of the node with maximum length m. The coordination problem is to consider these n nodes' information and make a series of sequential decision, to maximize the data throughput. To model it as a MDP problem, we need to define the tuple of (S, A, P, R) here:

State: $S = \{\triangle Q, Q_{HP}, L_i, L_2, \ldots L_n\}$. Here, $\triangle Q$ represents the $n \times m$ matrix which contains the deduced sequence number of each packet. Q_{HP} represents the sequence number of the HP. $L_i, L_2, \ldots L_n$ represent the queue lengths of each flow.

Action ($a = a_1, a_2, a_3, a_4 \in \{0, 1, \ldots, n\}$ and $a_1 > a_2 > a_3 > a_4$): The decision is to let a certain node i transmit with the HP. Since at most 5 nodes [50] can send packets simultaneously, the total dimension of a is four. $a_i = 0$ means choosing none of the nodes, at the i^{th} dimension.

Reward $R(t)$: This equals the total throughput achieved, i.e. the aggregate throughput from AP to all the destination node, during the coordination period t. The AP knows the aggregate throughput by hearing the ACK from the destination node.

The state transition model P is dependent on the packet incoming rate and the channel condition. If we know this value, then P can be calculated offline. However, normally, these quantities are difficult to obtain. Luckily, the MDP can still be solved online (interacting with the real world) by a model free Reinforcement Learning method, as long as the probability is fixed. It means that P can be unknown.

Simplify the MDP by merging state

The above MDP formulation for the scheduling captures all the details of the state transition. It can give the optimal performance, if we can successfully solve it. However, the problem is, it is practically unsolvable. This is because the state space has an extremely large dimension (more than $m \times n$) and the state space is continuous, since the sequence number can be a continuous value. As shown in [51] and [52] the computational complexity for such cases, increases exponentially with the dimension of the state space. In order to make the problem practically solvable, we have to reduce the state space, and preferably make it a discrete value. Meanwhile the state space should still be able to regulate the action selection and capture the major factors of the information to facilitate the action selection.

With these goals in mind, we would like to design the state space as $S = W_1(t), W_2(t), \ldots W_n(t)$, where $W_i(t)$ equals the number of packets, which have the sequence number less than $Q_{HP}(t) + \tau$, in node i at time t. It means, the value of W_i is always an integer number with $0 \leq W_i \leq [\frac{\tau}{Psize_i}]$, where $[\cdot]$ represents the floor function. The overall number of states (not the state dimension in previous case) is: $\prod_{i=1}^{n}[\frac{\tau}{Psize_i}+1]$. This makes the solving process much easier. The new state space can be viewed as mapping from the previous state space with a merging process. Since all the states in the previous state space can be mapped to one and only one new state. Hence, it still captures all the Markovian properties. Although the optimal solution for this model may not be as good as the previous one which captures all the details, we believe that the state space still gives a good indication of

how the action should be chosen. After the state merging, we have modeled the problem as a discrete finite state MDP, Most importantly, the merging process makes the problem practically solvable.

Solve the MDP by a RL algorithm

There are many RL algorithms, which can solve the given discrete finite state MDP problem. The one we choose in SI-CCMAC is the well-known actor-critic algorithm [38]. It uses the temporal-difference (TD) techniques, which will be explained in more details later. One advantage of TD techniques is that, they can learn directly from raw experience without a model of the environment's dynamics. Hence it is a model free algorithm, which is suitable for the given MDP problem.

There are two main components of an actor-critic system: the actor and the critic. The policy structure is known as the actor, because it is used to select actions. The estimated value function is known as the critic, because it criticizes the actions made by the actor. The learning process is done by changing the policy the actor follows, based on the feedback (critique) from the critic. The critique is the TD, δ_t, calculated in time t, which is a scalar value and can be represented as:

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{4.3}$$

The r_{t+1} is the reward received at the next step, γ is the discount factor and V is the current value function, similar as the Q-value [38], at different states. The value function is related with the current policy. As the name suggests,

it represents the expected sum of rewards to be received in the future, with the given current state and under the current policy π. Mathematically, it can be represented recursively as:

$$V^{\pi}(s) := E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t = s\} \tag{4.4}$$

In real life, getting the actual value function is virtually impossible without knowing all the transition probabilities. A simple way to estimate the value function is to use the following equation in each step,

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1})] \tag{4.5}$$

where α is a constant step size parameter.

From the Equation 4.3, it is clear that, the TD is the difference between the total rewards which will be received after executing the action, $r_{t+1} + \gamma V(s_{t+1})$, and the expectation of future rewards before the action selection, $V(s_t)$. Hence, it is a proper metric to evaluate the action just selected. If the TD error is positive, it suggests that the tendency to select the previous action at the previous state should be strengthened for the future, otherwise if the TD error is negative, it suggests that the tendency should be weakened.

Lastly, the actor-critic method handles the feed back and choose the action by using the Gibbs softmax method:

$$\pi_t(s, a) = Pr\{a_t = a | s_t = s\} = \frac{e^{P(s,a)}}{\sum_b e^{P(s,b)}}, \tag{4.6}$$

where the $p(s, a)$ are the values at time t of the modifiable policy parameters

of the actor, indicating the tendency to select each action a when in each state s. Then the strengthening or weakening process, described above can be implemented by increasing or decreasing $p(s, a)$ by

$$p(s_t, a_t) \leftarrow p(s_t, a_t) + \alpha\delta_t, \tag{4.7}$$

where α is another step size parameter.

4.3 Simulations and Results

4.3.1 Simulation Setup

The simulator we use here is still OMNet++ [41]. The network setup is similar as the simulation of CCMAC, where an AP is at the center of a unit circle with radius equal to 100 m. Two types of nodes are scattered in two different areas. Relay nodes are uniformly and independently distributed in the inner circle of radius equal to 67.1 m. Destination nodes are uniformly and independently distributed in the outer ring, with inner radius equal to 67.1 m and outer radius equal to 100 m. The channel loss is modeled by two factors which are the bit error rate (BER), and the channel fading process. The BER is set as 1×10^{-4}. The fading process is modeled as a two state Markov chain [42] [43], either the link is "up" or "down". The probability of a link going down is p_1, and the probability of a link going up is p_2. In this simulation, we let $p_1 = 0.05$, $p_2 = 0.2$ and the duration for each state is a uniformly distributed random number between $0 - 2$ seconds.

We study the performance of five different protocols: WiFi, CoopMAC,

SI-CCMAC$_{\text{mdp}}$, SI-CCMAC$_{\text{mis}}$, and random$_4$. The WiFi is the legacy 802.11b DCF protocol with channel rate adaptation, i.e. the RBAR extension. CoopMAC is the protocol described in [5], and is the main comparison target for SI-CCMAC. SI-CCMAC$_{\text{mdp}}$ is the protocol proposed in this chapter with a MDP modeling. The SI-CCMAC$_{\text{mis}}$ protocol finds out the maximum independent set among the unselected nodes. The protocols of random$_4$ have the same cooperation part as SI-CCMAC. However, these protocols do not perform coordination. It randomly chooses 4 other nodes to send packets together with the HP, regardless of whether such nodes may collide with each other or not.

4.3.2 Experiments

In the first experiment, there are 8 senders randomly scattered in the outer ring area, and the packet size is 1024 bytes. We vary the number of relay nodes to see the performance of different protocols. We repeat these experiments for 10 different topologies. The results of average throughput performance are shown in Figure 4.4. The performance of WiFi is not affected by the number of relay nodes, since it does not use relaying. Hence, it gives a flat line. All the other protocols benefit when there are more relay nodes. We find that SI-CCMAC has significant advantages over WiFi and CoopMAC. The advantage increases when more nodes join the network. Even the non-coordinated random$_4$ algorithms can out-perform WiFi and coopMAC. Between SI-CCMAC$_{\text{mdp}}$ and SI-CCMAC$_{\text{mis}}$, the MDP version performs slightly better than the MIS version. However the advantage is

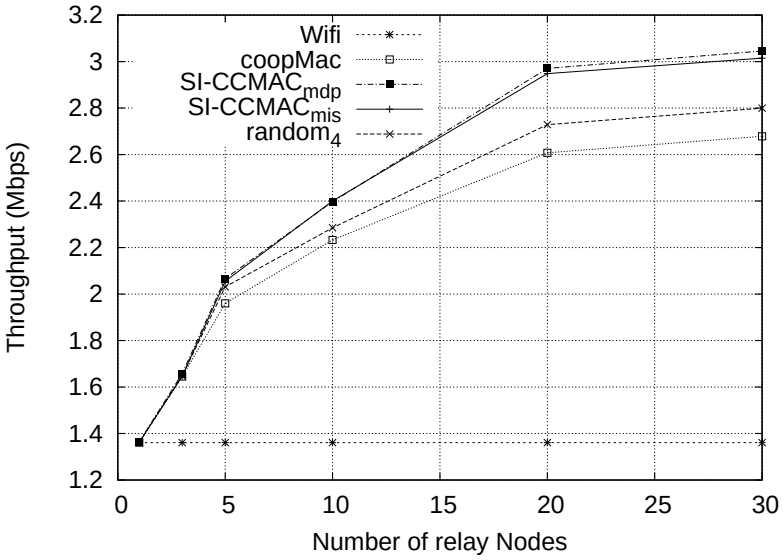


Figure 4.4: The average throughput achieved with different number of relay nodes.

marginal. In addition, the MIS based coordination scheme is computationally much cheaper than its counter part. Hence, we believe that the MIS modeling has good performance and is practically preferable.

In the second experiment, we set the number of the relay nodes to 20 and the packet size as 1024 bytes. We randomly add in more sender nodes and see the performance difference. We repeat these experiments for 10 different topologies. The average results are shown in Figure 4.5. We also randomly pick 3 sets of result from these 10 individual experiments, which are shown in Figure 4.6(a), Figure 4.6(b), and Figure 4.6(c). From all the graphs, we can see that SI-CCMAC is consistently better than the other two protocols. Clearly, with more destination nodes, SI-CCMAC achieves better performance. The reason is that, with more destination nodes, it may be able to choose more nodes to transmit simultaneously. Between the SI-

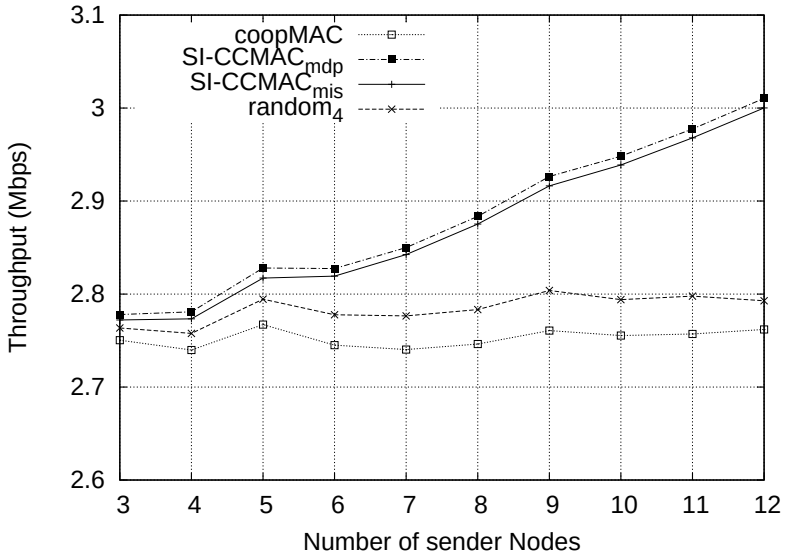
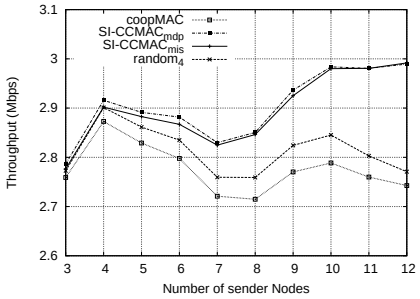


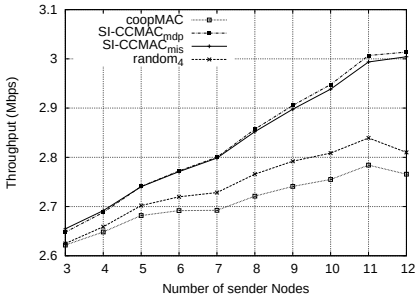
Figure 4.5: The average throughput achieved with different number of sender nodes.

$\text{CCMAC}_{\text{mdp}}$ and $\text{SI-CCMAC}_{\text{mis}}$, the result shows, both of them have very similar performance and no one has significant advantages over the other.

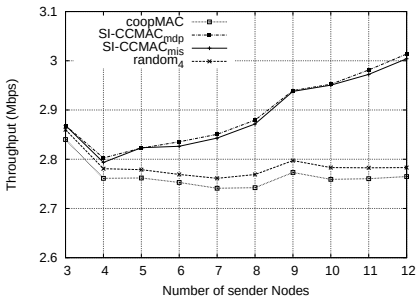
By combining these two experiment results, we find that SI-CCMAC has significant advantages over both WiFi and CoopMAC for nodes with poor direct transmission link. In addition such advantages increases when more nodes join the network. Even the non-coordinated random_4 algorithms can slightly outperform WiFi and coopMAC. Between the $\text{SI-CCMAC}_{\text{mdp}}$ and $\text{SI-CCMAC}_{\text{mis}}$, theoretically speaking, $\text{SI-CCMAC}_{\text{mis}}$ should have advantages over the $\text{SI-CCMAC}_{\text{mdp}}$, because the MDP consider long term optimization while the MIS just consider one step. However, the simulation results show that, such the advantage is insignificant. In addition, coordination using the greedy method is computationally much cheaper. Hence, we believe that the MIS based method is practically more preferable than



(a) Case One.



(b) Case Two.



(c) Case Three.

Figure 4.6: The throughput performance based on 3 different network topologies.

the one based on MDP.

4.4 Conclusion

In this chapter, we explored the benefits of cooperation and concurrent transmissions at the downlink of wireless LAN. We proposed a novel sender initiated concurrent cooperative MAC (SI-CCMAC) protocol which utilizes these features to improve the throughput performance of the network. SI-CCMAC has three different transmission modes: direct mode, two-hop

mode, and multi-destination mode. Based on the decision given by the downlink allocation algorithm, after relaxing the fairness constrain, this allocation problem is modeled as a vertex coloring problem, maximum independent set problem and a MDP problem based on different scenarios. Existing algorithms can then be applied. Through simulation, we find that, the MIS modeling is both widely applicable and has relatively good performance. Hence, we recommend this method.

After studying the MAC layer of wireless networks, we will move to a higher layer. Issues of opportunistic cooperation is studied at the network layer, in the next chapter.

Chapter 5

Geographic Multi-hop-Sift

In wireless sensor networks(WSN), traditional network routing algorithms are challenged by nodes' propensities to go to sleep, move around, or even break down. It is costly in terms of communication and energy consumption for routing information to be kept up-to-date. Motivated by the recently proposed sensor network MAC protocol Sift [53] and the idea of geographic opportunistic forwarding, in this chapter, we propose a new hybrid opportunistic forwarding protocol: Geographic Multihop-Sift (GMS). It combines two forwarding techniques: priority list and random access. GMS is designed to be both energy efficient and robust against channel fluctuations or frequent changes of network topology. The performance of GMS is compared to two well known geographic routing protocols, GPSR [54] and GeRaF [11]. The simulation results show that GMS has advantages, in terms of delay, packet loss and energy consumption, in various scenarios.

5.1 Introduction

A typical wireless sensor network(WSN) has a high density of sensor nodes which send data intermittently towards a common sink or destination node. Due to the shared wireless communication medium, nodes which want to transmit data in a neighborhood must do so in a way that minimizes collisions among the nodes. A medium access control (MAC) protocol such as IEEE 802.11 [55], S-MAC [56] or Sift [53] would usually take care of this link-level communication.

In the case of packet forwarding or routing from a source sensor node to a destination node, in what is essentially an infrastructureless wireless ad hoc or multi-hop network, intermediate nodes are required to relay data packets on behalf of source sensor nodes which are out of range of the destination. A number of routing protocols for wireless ad hoc or multi-hop networks such as DSR [57] and AODV [58] have been developed, but these protocols are unsuitable for wireless sensor networks with resource constraints, since they involve caching a significant amount of network state information and periodic exchange of a substantial amount of routing information. Recently, multi-hop routing protocols which have significantly lower overheads, and are thus suitable to be used in sensor networks, have been developed, such as GPSR [54] and GEAR [59]. However, these routing protocols require the sender node to select one node among its neighbors to forward its packets, which implies that the sender node needs to maintain state information about, at least, its surrounding nodes in order to be able to make this choice.

Consider an event driven WSN, such as, a network for fire detection or target tracking. The network traffic is bursty. There may not be any network traffic for a long time; followed by a burst of packets sent continuously from a few sensor nodes to the sink. In such a network, the energy conservation and the delay of the packet forwarding are the major concerns. Protocols requiring frequent signaling to update the routing information between neighbors, may not be optimal with respect to these metrics, since they consume a significant amount of energy while the network is idle.

Being aware of such requirements, we propose a novel packet forwarding protocol for wireless sensor networks referred to as the *Geographic Multi-hop Sift* (GMS) protocol. To use GMS, we assume that nodes know their own location information before hand. One important feature of GMS is that the senders do not need to maintain state information of their neighbors. Instead, the sender node simply broadcasts a forwarding request message to the network, optionally specifying a list of preferred nodes (LPN). Its neighbors which are able to help forwarding the packet will respond. This protocol will be described in detail in section 5.3.

Similar approaches, which are known as opportunistic routing, have been proposed, such as GeRaF [11] BLR [60], IGF [61] and BGR [12]. They can be divided into two categories: the relay nodes are opportunistically chosen by the sender node or by the neighbors of the sender node. In the first category, sender nodes still require the state information of their neighbors, which is hard for resource-constrained sensor nodes to store and maintain. In the second category, the performance is affected by a large number of

duplicate packets, generated by over-zealous or uncoordinated neighboring nodes, which leads to many packet collisions and packet duplications, and consequently longer delays and lower efficiency. We will discuss these issues in detail in section 5.2.

Our contributions in this chapter are:

1. a hybrid opportunistic packet forwarding protocol, Geographic Multi-hop-Sift (GMS), which uses the LPN and geographic-Sift distribution to determine which node is selected to forward the packet for the source node. GMS is lightweight in terms of the signaling required, yet robust in the presence of frequent network topology change and unreliable wireless channel;

2. analysis which shows that the proposed scheme is optimal in terms of maximizing the rewards related to the progress being made to the destination with each transmission;

3. detailed comparison with the well-known GPSR and GeRaF protocols. Simulation results show that GMS performs better in the presence of fading channels and random sleep and wake processes in wireless sensor networks.

5.2 Packet Forwarding in Wireless Sensor Networks

In a WSN, nodes often go to sleep to conserve energy and, like other wireless networks, the links between nearby nodes are not continuously reliable. It is costly for the sender to know and update such state information of

its neighbors. Sometimes, the energy consumed by such overhead may be more than the actual packet forwarding process, when the network traffic is low. Hence, in such cases, the traditional "best" path routing/forwarding approach do not perform well.

One alternative approach, known as opportunistic routing, chooses the routing path on the fly. By having multiple relay candidates (next hop nodes), performance is improved because there is a higher probability that one of the nodes is awake and reachable. There are some opportunistic routing protocols proposed such as AnyCast [7], EXOR [8], SDF [9], GeRaF [11], IGF [61], ROMER [13]. In our view, these protocols can be divided into two different categories.

In the first category, some of the neighbors are selected as the relay candidates by the senders and only those sender-recognized relay candidates may participate in the forwarding procedure. Protocols, such as SDF [9], AnyCast [7] and EXOR [8] are in this category. In SDF, the sender broadcasts a RTS packet, which includes a list of relay candidates, to its neighbors. Candidate nodes, which successfully receive the RTS packet, send back the response packets following the order of candidate list. Then, among the responding nodes, the sender chooses the relay node with the most forward progress. In [10], a more efficient protocol was proposed, in which the sender keeps track of the performance among all the candidates and sends a priority list in the RTS. As the candidates respond in the order of the list, the sender immediately starts transmitting the data to the first responding candidate, once the first response packet is received. By doing this, it reduces the over-

head associated with waiting for multiple replies. The protocol EXOR [8] further extended the idea and improved the efficiency by employing a *batch* mechanism. In this protocol, a number of packets are forwarded at one time. By overhearing the progress made by other relay nodes, candidates help to relay the packets which have not been relayed yet.

In the second category, the senders are not required to have any knowledge of their neighbors. Instead, one or multiple relay nodes will be automatically selected among the neighbors. Protocols such as GeRaF [11], BGR [12] and ROMER [13] are considered within this category. The basic idea of GeRaF and BGR are quite similar. Basically, the sender broadcasts a request to the network, all the neighbors which heard this request would send back their reply after a certain amount of delay. Then the first responding node is selected as the relay node. The delay time is determined by the neighbors themselves and all the neighbors have a chance to participate. The protocol of ROMER uses multi-cast techniques to relay data. The sender broadcasts the data with a requirement of maximum delay it allows; all the neighbors which fulfill the requirement will further broadcast the data.

5.2.1 Problems of existing opportunistic forwarding protocol in WSN

Although protocols in both categories have good performance when link or node failures happen frequently, they have several shortcomings.

For the protocols within the first category, firstly, nodes need to know

their neighbor's state information (may not be very accurate), then pick relay candidates before packet transmission. Hence, periodic signaling between neighbors may still be essential, although the frequency can be lower than those of the "best" path algorithms. Secondly, since a dedicated time slot is given to each relay candidate to respond, it may cause large delay, when most of these selected candidates are not available. Lastly, there is a trade-off between the performance and the complexity. Suppose there are M nodes inside the network and, on average, N relay candidates are specified for each packet forwarding. Nodes may need to maintain, up to, $M-1$ priority lists with the length of about N nodes. In addition, the nodes have to broadcast the request with, in average, N nodes' ID each time. As a result, in order not to make the protocol too complicated, the N cannot be large (in [9], the author suggest $2 \leq N \leq 4$). On the other hand, in order to acquire higher channel/space diversity and to be more robust, a larger N is required, which is suggested in EXOR.

Let us look at the problems that arise in protocols in the second category. Firstly, because of the contention based (random access) technique, there will be large delay during the selection of relay nodes. Secondly, again due to the contentions, there are frequent packet collisions and duplications. Protocols like GeRaF, BGR, have their own ways of resolving the collision, either by contentions or using additional random delays. However it causes a lot of extra cost including extra network load, delay and energy. Hence, efficiency suffers.

5.2.2 A hybrid solution given by GMS

Taking into account the advantages and disadvantages of different kinds of protocols, we propose our protocol: Geographic Multi-hop-Sift (GMS). It combines the features from both the categories and mitigate their disadvantages. The details of the protocols are introduced in Section 5.3. The core idea is as follows: the sender broadcasts a request, which optionally contains a list of preferred nodes (LPN), to all the neighbors (with positive progress) and waits for them to reply. The first node, replying the request and being heard by the sender is chosen as the relay node. Among all neighbors, the LPN nodes are given dedicated time slot in the beginning. If none of the LPN nodes response, the other neighbors choose their own time slot by the geographic-Sift distribution. The nodes in LPN are not static. They are the winner of the previous open competition (by geographic-Sift). The geographic-Sift is an extension of Sift distribution [53], which was designed to minimize the probability of packet collisions in such scenarios.

With the LPN and the geographic-Sift distribution, there are no requirement of periodic signaling between nodes and the collision of response packets will also be very much minimized. And due to the LPN, the delay overhead of selecting the relay node is also small. In addition, since all neighbors has a chance to be selected, the protocol is robust even if most nodes are not available.

5.3 The Geographic Multi-hop Sift (GMS) protocol

Taking into account the advantages and disadvantages of the protocols described in section 5.2, we propose a hybrid opportunistic protocol: Geographic Multi-hop-Sift (GMS). It uses two techniques: LPN and geographic-Sift. In this section, we will first introduce the LPN technique (the geographic-Sift is introduced in next section), then give the details of how to combine these two techniques and hence how the GMS works.

5.3.1 Determining the LPN

The LPN is a memory based technique, which requires little signal exchanges and energy consumption. There are two parts of this technique: the node selection and list construction.

The node is enrolled in the list, after it has been selected as relay node through the open competition, i.e.the geographic-Sift. This means the node is the "best" node, which is currently available. There are two tags associated with each node. The first is a time to live (TTL) tag which shows how long the node may still be available, i.e. the time before the node goes to sleep again. Normally, the value of TTL is known by the relay node itself. The second is the utility tag. This utility can be any metric to be optimized, for example: node's residual energy, distance from the sink, as long as the metric does not change within a short time period. In addition, this utility can also be a linear combination of a few selected metrics. We assume that the utility is also known by the relay node itself. In GMS we concentrate

on finding the node with the largest distance progress. Hence the utility is defined as the projected local distance gain. In order to let the sender know the value of the tags, the relay node needs to piggyback them in the CTS message.

After the nodes have been selected, they are put into the LPN. Nodes are arranged by the value of their utility, which means higher priority is given to the node with higher utility. For all nodes, there is a maximum life time T. Nodes are removed from the LPN, if their TTL expire.

There are three points to be taken care of. Firstly, the LPN can be empty. This normally happens when the sender just wakes up. In such cases, there is no time slot reserved for LPN. Secondly, the tag is always updated by the new value. Through this, we allow nodes to change their TTL and utility. Lastly, for nodes which are one hop away from the destination, the highest priority of the LPN is always given to the destination node.

5.3.2 GMS: Basic operation

In our proposed scheme, a sender node sends a request-to-send (RTS) message to the network, which may contain a list of preferred nodes (LPN), and wait for one of the nodes within its transmission range to, in effect, reply "please forward the packet to me", indicated using a kind of Clear To Send (CTS) message. The first node to send back the CTS and be heard by the sender is chosen as the relay node. Hence, the delay of neighbors sending the CTS after receiving the sender's RTS is critical in determining the next-hop node.

Consider the case where node A has broadcast an RTS which specifies node B as the only node in LPN. Many neighboring nodes, including node C, have received this request. All nodes which hear the RTS are synchronized by this event, and start counting time slots at the same instant of time.

Contention phase for claiming the CTS

Each node which hears the RTS immediately calculates a time slot in which it will send the CTS. The first slot is reserved for node B. All the other nodes calculate their respective time slot as 1 plus the result from the geographic-Sift distribution, as described in section 5.4.2; and one Bernoulli random slot to prevent persistent CTS collision. Until this time slot arrives, each node listens for any event, indicating that another node has claimed the right to receive and forward the packet.

The events are: (1) Node hears the data packet from node A; (2) Node hears a CTS from some other node. Suppose these events happen at node C. Event (1) implies that some other hidden node has already sent a CTS back to A, and now A is sending the data. Under such cases, node C should drop his CTS packet immediately. Event (2) implies that some other node has sent a CTS. It forces node C to keep silent for some number of slots defined in the CTS, corresponding to the number of slots required by A to send the data packet. If node C originally wanted to send a CTS within this silent period, the Sift game is considered over as far as node C is concerned, and therefore node C simply drops its own CTS.

However, if none of these events has occurred by the time node C's slot arrives, node C sends a CTS to claim the right to receive the data packet

from A. If all goes well, node C will then receive the data packet from node A.

If another node happens to choose the same time slot as node C, then a CTS collision will result. Node A will receive a corrupted packet and the transmission attempt ends in failure. However, if node B hears the RTS and is willing to relay, it will always be the first responding node without any danger of CTS collision; hence, node B will win the game in most of the cases.

Receiving and acknowledging the data packet

Given that node A has received a CTS from node B, it sends the data packet addressed to node B. Node A then waits to hear an implicit acknowledgment from node B. If node B is not successful in receiving the data packet, it simply waits for node A to time out and retransmit. Once it receives the data packet successfully, it sends an RTS into the network to discover the next relay node. The sending of the RTS also implicitly acknowledges the successful reception by B of the data packet from node A.

5.3.3 Packet retransmission

If node A successfully receives an implicit acknowledgment from a relay node R, it will end node A's responsibility for that data packet. However, if A did not receive the acknowledgment, A will believe that the packet is lost and decide to retransmit this packet. However, this time, A will put node R as the first node in LPN and indicate that it is a retransmission of the last packet in the RTS. If node R hears this RTS and discovers that the

packet had already been received, an acknowledgment will be directly sent back to A using the reserved slot; otherwise, it follows the rules of normal packet forwarding. By doing this, i.e. giving the "failed" relay node a second chance, it reduces the possibility of packet duplication and saves energy.

5.3.4 Recovery phase

Although we assume that the node density of the network is high, it is still possible that, there is a big obstacle or in the network, so that a node cannot find any neighbors closer than itself to the destination. Such situation is detected when none of the neighbors answer the sending request for a consecutive k times. In such case, we cannot use the greedy routing but have to enter the recovery phase. Here, we apply the right-hand rule techniques [54] to solve such problem. It means the packet is forwarded along the contour of the region which have no nodes in a counterclockwise manner. Hence, we can eventually route around the obstacle and back to the greedy phase.

5.4 The Sift and Geographic-Sift Distribution

5.4.1 The Sift distribution

The Sift protocol itself is a non-persistent CSMA protocol for the MAC layer of a WSN. It is a slotted protocol in which an attempt is made to resolve, over a fixed time window of W slots, contention for the channel that has arisen from N nodes simultaneously sensing the same event. The objective of Sift is to maximize the probability that one of these nodes is successful.

Each sensor node independently selects a time slot in the range $1, 2, \ldots, W$ according to an increasing, truncated geometric distribution:

$$\begin{aligned} p_r &= \tfrac{(1-\alpha)\alpha^W}{1-\alpha^W}\alpha^{-r} \qquad r = 1, 2, \ldots, W-1 \\ p_W &= 1 - \textstyle\sum_{r=1}^{W-1} p_r \qquad r = W \end{aligned} \tag{5.1}$$

where $0 < \alpha < 1$ is a chosen parameter. This distribution is termed the "Sift distribution" in [53], and we shall also use this terminology. It is shown in [53] that a near-optimal choice of α over a wide range of population sizes, N, is $\alpha = M^{-1/(W-1)}$, where M is the maximum over the considered population sizes.

The above distribution is such that most nodes choose later slots, and only a few, if any, choose early slots. We can think of the protocol as a game in which a sensor wins in slot r if it is the only sensor to pick slot r, and all other nodes pick later slots. If a single winner emerges at slot r then the game immediately terminates, with the remaining sensor nodes giving up. If, instead, more than one node transmits at the same time, then there is a collision, and the game terminates without a winner. The objective of Sift is to approximately maximize the probability of obtaining a winner.

5.4.2 The geographic-Sift distribution

Now let us adapt the Sift MAC layer protocol to the routing layer problem of packet forwarding. The similarity between the two problems is clear: in the forwarding problem, the equivalent of the "sensed data" is the "request received", the equivalent of the N sensor nodes are the N wireless nodes

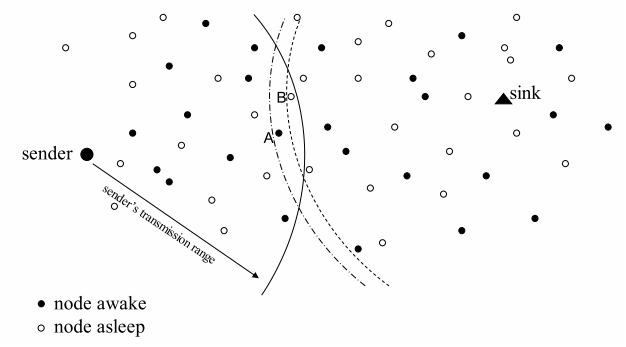


Figure 5.1: Geographic Multi-hop Sift (GMS) operating scenario: sender, sink and potential forwarding nodes.

that have heard the request, the equivalent of the winner is the node that is selected by the network to provide the next hop for the packet. As above, N is random, and unknown to the sender of the request.

The key difference here is that we wish the network to perform optimization. It is not enough to just select a winner, but the winner must be a good choice in terms of progress of the data packet towards the destination.

Let us take the Sift setup, with W slots available to resolve the contention for the channel by the nodes that have heard the request. Let each node i that hears the request and participates in the Sift game be allocated a reward R_i, that represents a measure of progress toward the sink. We wish to use the Sift strategy to pick the node i with the largest value of R_i. We assume that each node i can measure its own R_i, but does not know the rewards of the other contending nodes.

Optimization framework

The interval $[0, 1]$ is partitioned into $W+1$ disjoint sets $\mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_W, \mathcal{P}_{W+1}$. A strategy for choosing the winning node using the partition is to allow node

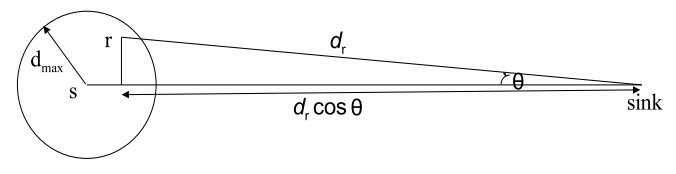


Figure 5.2: Approximation to obtain distribution of R.

i to select slot r precisely when $R_i \in \mathcal{P}_r$. Node i does not participate at all if $R_i \in \mathcal{P}_{W+1}$. The Sift game then proceeds as above: empty slots occur, until at some slot r there is a transmission of the response. If there is a single transmission, we have a winner, and otherwise a collision occurs.

This framework allows the formulation of an optimization problem. Let R denote the value of the winner, or 0 if there is a collision. If the R_i are modeled as continuous random variables in the unit interval $[0, 1]$, admitting density functions, a natural optimization problem to consider is that of maximizing $\mathbb{P}(R \neq \max_i R_i)$. The following lemma applies:

Lemma 5.1 *The partition $\mathcal{P}^*$ that solves the problem:*

$$\max_{\mathcal{P}} \mathbb{P}(R \neq \max_i R_i)$$

has the following form:

$$\mathcal{P}_r^* = (d_r, d_{r-1}] \quad r = 1, 2, \ldots, W$$

for some decreasing sequence $(d_r)_1^W$, with $d_0 = 1$.

Proof: For any partition, $\mathcal{P}$, we have that

$$\begin{aligned}\mathbb{P}(R \neq \max_i R_i) &= \mathbb{P}(R = 0, \max_i R_i > 0) \\ &\quad + \mathbb{P}(R > 0, R \neq \max_i R_i)\end{aligned}$$

Note that any partition $\mathcal{P}$ can be replaced with a partition of the form stated in the lemma, with no change to the first term on the right hand side. However, this replacement results in zero for the second term on the right hand side. This observation can be applied to the optimal $\mathcal{P}^*$ yielding the result. ∎

The lemma also shows that even a suboptimal partition is "locally optimum" if it has the above structure, in the sense that the only way to improve the objective is to improve the sizes of the intervals (the Sift probabilities); no improvement can be made by permuting the allocation of intervals to time slots. We will indeed use a suboptimal partition defined in such a way as to obtain the probabilities from the truncated geometric "Sift" distribution. This will ensure that collisions are kept to a minimum. By choosing the unique partition with the above structure we will ensure that the winner is the the node that has the highest value of R_i provided that there is no collision.

Example: uniformly distributed rewards

For simplicity, let us first consider the case of independently and uniformly distributed rewards. There are a random number, N, of nodes, and each R_i is uniformly distributed on $[0, 1]$.

In this example, we calculate the d_r's corresponding to the Sift truncated geometric distribution. We will use a partition of the above form, and node i will transmit in the slot r for which $R_i \in (d_r, d_{r-1}]$. The problem then is to choose the d_r's to obtain the increasing, truncated Sift distribution. The following lemma contains the result.

Lemma 5.2 *Choose the sequence* $(d_r)_{r=1}^W$ *according to:*

$$d_r = \prod_{j=1}^{r} x_j \quad r = 1, 2, \ldots W \tag{5.2}$$

where

$$x_r = \frac{\sum_{i=0}^{W-r} \alpha^i}{\sum_{i=0}^{W-r+1} \alpha^i} \quad r = 1, 2, \ldots, W.$$

Randomly select the random variable R *from the* $U[0,1]$ *distribution, and select slot* r *iff* $R \in (d_r, d_{r-1})$. *The induced probability mass function on slots is provided in (5.1).*

Proof: Let p'_r denote the conditional probability that slot r is chosen by node i, given that the node did not choose any prior slot (in the range $1, 2, \ldots r-1$). Then the geometric nature of the Sift distribution implies that

$$1 - p'_r = \frac{1 - \alpha^{W-r}}{1 - \alpha^{W+1-r}}.$$

This is straightforward and can be shown by induction. Now assume that the first $r-1$ intervals above do generate the first $r-1$ Sift probabilities. Using the fact that if we condition R_i on the interval $[0, d_{r-1})$, it is conditionally

uniform on that interval, it follows that if

$$x_r = 1 - p'_r$$

then the first r intervals do generate the first r Sift probabilities. The result follows by induction. ∎

The lemma defines the partition to use when the R_i are uniformly distributed on $[0, 1]$.

General case: arbitrarily (but independently) distributed rewards

In the general case, if the nodes know the distribution function of the rewards, F, then transformed rewards, U_i, can be generated via $U_i \equiv F(R_i)$. It is well known (and trivial to show) that the U_i are independent and uniform random variables on $[0, 1]$. Maximizing the R_i is equivalent to maximizing the U_i, so the nodes can use the d_r's computed in section 5.4.2 to compute the slot to send their CTS back to the sender; node i simply replaces R_i by U_i in the computation.

In practice, it is probably not reasonable to assume that the nodes have knowledge of the distribution function F. In the spirit of Sift, we seek a suboptimal but robust choice; in particular, a fixed distribution function, independent of all other parameters.

Assume that there is a maximum transmission range, $d_{\max}$, beyond which no node can hear the RTS from the sender (or simply a distance beyond which no node is allowed to participate in the Sift game). Let d_s be the distance from the sending node to the sink. Consider a circle of radius d_{max}

surrounding the sending node, and consider any potential relay node within this circle. Let d_r be the distance from the relay node to the sink. A natural measure of reward is the normalized value $R = (d_s - d_r)/d_{\max}$, which lies in the unit interval $[0, 1]$ (we only allow the possibility of forward progress in the half circle around the sender directed towards the sink, i.e. we discard any node for which the measure is negative).

Now consider the distribution of R. The exact distribution will be a function of the location of the sending node, relative to the sink, and we want to avoid this feature in a practical implementation. If we approximate the situation by assuming that the sender is far from the sink, then d_r is approximately equal to the distance from the sink to the point that is obtained by projecting the position of the relay node onto the line joining the sender to the sink. See Figure 5.2 for a depiction of this scenario; if the sender is far from the sink, the angle θ depicted in the figure is small, so $d_r \cos\theta \approx d_r$. Now if the listening nodes are assumed to be uniformly distributed in the circle of radius $d_{\max}$ around the sending node, then the density of the projected distance is approximately linearly decreasing with $d_s - d_r$. In other words, the probability density is highest at projected distance equal to d_s, and lowest (equal to zero) at projected distance equal to $d_{\max}$; it decreases approximately linearly to zero, as we decrease d_r from d_s to $d_s - d_{\max}$.

Translating this argument into normalized values, we obtain the approximate density function $2(1 - R)$ for R, with distribution function $F(R) = 1 - (1 - R)^2$.

The derivation above, which we shall refer to as the "distance-to-sink-modulated Sift distribution" or "geographic Sift distribution", provides us with a theoretically sound method for determining which of a number of neighboring nodes should forward a packet. In the following section, we define the Geographic Multi-hop Sift (GMS) forwarding protocol. In this protocol, a node that receives the request will compute its corresponding normalized R value, insert this into $F(R) = 1-(1-R)^2$, and then determine the slot to send the response, using the Sift interval in which $F(R)$ lies.

5.5 Simulation Scenarios and Results

5.5.1 Network topology

The simulation tool we have used to evaluate the GMS protocol is OMNet++. In this simulation, there are N nodes uniformly scattered in a 400 m by 400 m 2-D square. In addition, there is a sink node at the center of the square. Each node generates packets and sends packets towards the sink. All of them have the same transmission range r m, which means that a node can only hear other nodes which are inside a circle of radius r (we set $r = 75$ m). However, it is not guaranteed that every node inside the circle can be heard, due to channel loss, the sleep-wake processes, and fading events. The channel loss probability (bit error rate) is fixed at (1^{-4}) for all simulation. the process of sleep-wake and fading is describe next.

5.5.2 Sleep and wake process (SWP)

A random sleep and wake process (SWP) is used in the simulation, and it applies to all nodes in the network except the sink. This SWP is a random process with a fixed duty cycle α, where $0 < \alpha < 1$. Once the SWP is on, a node alternates between being awake for a random period from 0 to 10α seconds, and then being asleep for a random period from 0 to $10(1 - \alpha)$ seconds. In each case, the random duration is chosen independently and uniformly.

5.5.3 Channel fading process

An independent two-state Markov chain [42, 43] is used to model the fading process for each wireless channel: either the link is 'up' ('good' state) or it is 'down' ('bad' state). For simplicity, we only allow a link to change state each time a transmission occurs. The probability of a link going down is p_1, and the probability of a link going up is p_2. By controlling p_1 and p_2, we can change the rate of fading, and also the average proportion of time the link is 'up'.

5.5.4 Packet generation and relaying

Fresh data packets are generated at each awake node in the network following a Poisson process with a constant rate which can be varied in different experiments. All packets are to be forwarded towards the sink. A time to live metric is set for each packet, which is 20 seconds from the time packets leave the source node. This means that if the packet cannot reach the sink

within 20 seconds, it will be discarded.

We compare our proposed GMS protocol with the Geographic Perimeter Stateless Routing (GPSR) protocol [54] and Geographic Random Forwarding (GeRaF) protocol [11]. GPSR is a sender-determined routing protocol which requires the sender node to make a routing decision and select the next hop relay node. GeRaF is an opportunistic forwarding protocol where the relay node is determined among the neighbors of the sender node.

For GPSR, every node needs to maintain a routing table and send out beacon packets containing information about its position. Beacon packets are sent out periodically with a uniformly-distributed interval $[0.5B, 1.5B]$, where B is the mean inter-beacon transmission interval (in our simulations, $B = 1.5$ seconds). Since GPSR is a pure routing protocol, we need a MAC protocol to handle the link layer-related issues such as packet loss and retransmission. In our simulations, we used the 802.11 MAC protocol, which is the MAC protocol used in [54].

5.5.5 Energy consumption

Since we want to compare the energy consumption among the three packet forwarding protocols, only the energy consumed during packet transmission and reception are considered. The value of transmission power and receiving power is copied from Crossbow MICA2 data sheet [62], which is $81mJ/s$ and $30mJ/s$ respectively. We do not consider the energy expended when the nodes are simply awake but idle, with no packet transmission or reception, since this depends on the sleep-wake process and not the packet forwarding

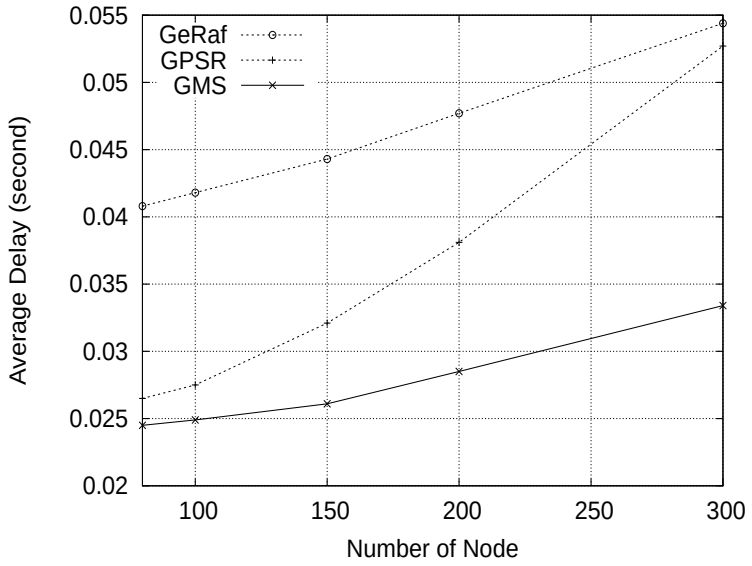
protocol. Although this makes the energy consumption graphs a little confusing when the SWP is on, it enables us to evaluate the energy efficiency of the packet forwarding protocols themselves, which is our main interest.

5.5.6 Experiments

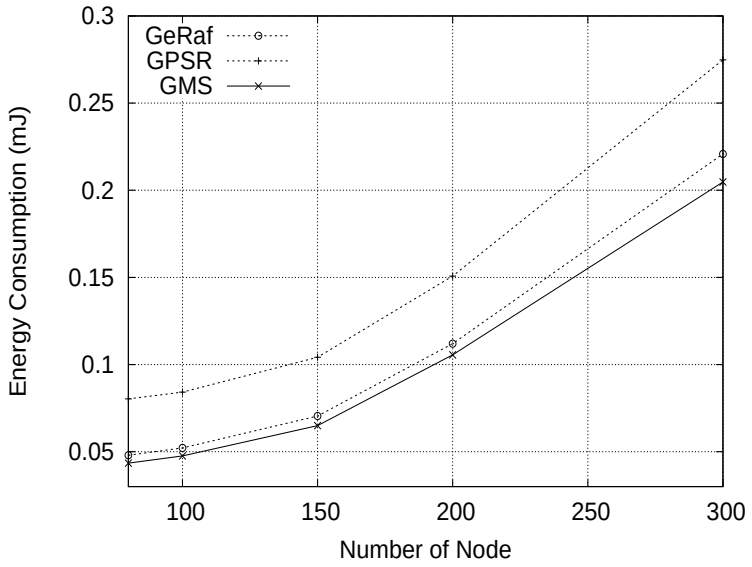
In this section, we consider four cases in which we compare the performance of GMS with GPSR and GeRaF, in terms of delay and energy consumption.

Case 1: Effect of node density

The first case has an SWP duty cycle of 1 and no fading. It means that nodes never sleep and network topology is stable. Figures 5.3(a) and 5.3(b) compare delay and energy consumption among GMS, GeRaF and GPSR when the number of nodes, i.e. the node density, is varied. Generally, GMS and GPSR have better delay performance than GeRaF when the node density is low. The reason is that, under such conditions, the relay node chosen using GPSR is the best node and is always reachable. The LPN in GMS is also targeted at the best node all the time. Hence, both protocols are always able to choose the best nodes and the delays incurred are low. In contrast, the node chosen by GeRaF sometimes is not the best; furthermore, GeRaF takes a longer time and more energy to find the relay node. When the node density increases, the delay of GPSR increases quickly. This is because, with more nodes in the network, there are more beacon packets that need to be sent within the network, which effectively causes more network load. On the other hand, the increase in node density has a smaller effect on GMS and GeRaF compared to GPSR.



(a) Delay versus node density.



(b) Energy consumption versus node density.

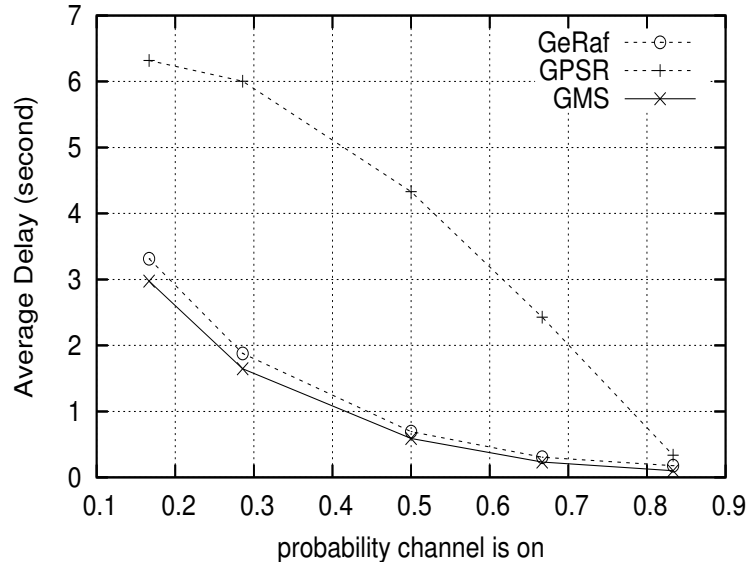
Figure 5.3: Case 1 (no fading, no SWP): delay and energy consumption.

Case 2: Effect of fading channel

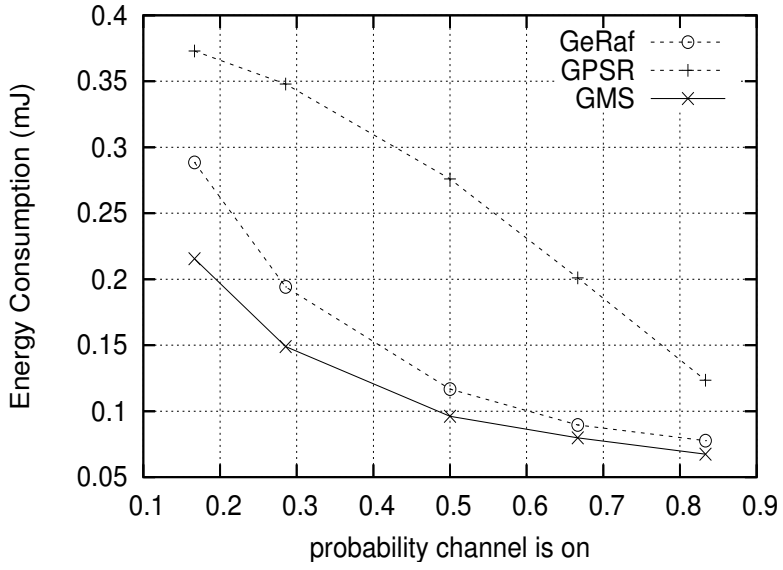
In the next scenario, we have a fixed number of 100 nodes randomly distributed in the field and a SWP duty cycle of 1. We vary the probability that the links are up ('good' state) to evaluate the performance of the different protocols. We do this by fixing p_1 at 0.05, and varying p_2, the probability of a link going from 'down' to 'up'. Figures 5.4(a) and 5.4(b) compare the delay and energy consumption of GPSR, GeRaF and GMS as the 'channel good' probability increases. Not surprisingly, GMS and GeRaF outperform GPSR when the 'channel good' probability is low. This is due to the fact that GPSR will often have out-of-date state information (unless the beacon rate is very high), while GMS and GeRaF use a decentralized method to find a relay node on the spot. When GMS is compared with GeRaF, GMS is better than GeRaF in terms of delay and energy used, which shows us that GMS is more robust and efficient than GeRaF.

Case 3: Effect of SWP duty cycle

Finally, we consider the case that nodes have a SWP duty cycle less than unity. In this case, there are 400 nodes in the network, and the value of p_1 and p_2 are fixed at 0.05. We have found that, GPSR performs much worse than the other two protocols. Because many times, they selected relay node have just gone to sleep, which cause long delay and large energy consumption. Hence, in this we just compare the performance between GMS and GeRaF. Figures 5.5(a) and 5.5(b), we observe that GMS wins substantially over GeRaF with respect to delay and energy consumption. It can be



(a) Delay versus channel state condition.



(b) Energy consumption versus channel state condition.

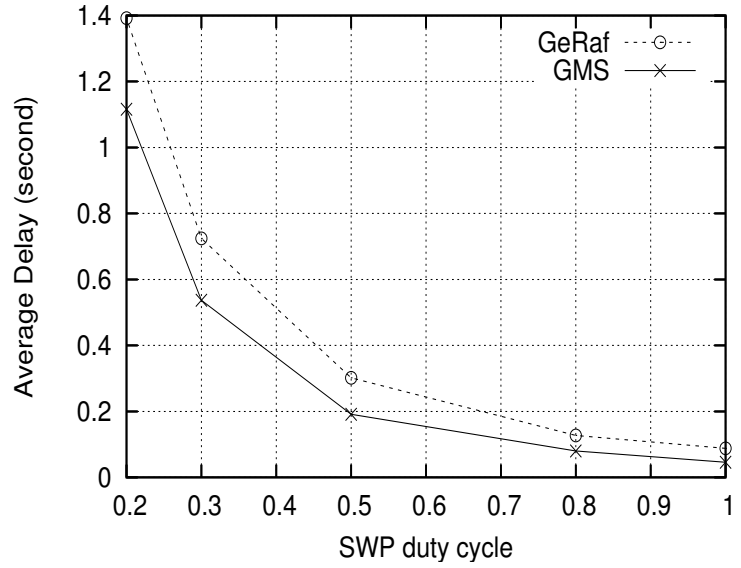
Figure 5.4: Case 2 (with fading, no SWP): delay, energy consumption and packet loss rate.

seen that, in order to achieve a particular delay or loss rate performance, GeRaF needs to operate at a higher SWP duty cycle than GMS, which shows that the GMS protocol is more energy efficient. In addition, it also shows again the robustness of GMS since the number of neighboring nodes and channel state keep on changing. As mentioned above, the energy graph, Figure 5.5(b), shows the energy consumption from packet transmission and reception, and excludes the energy used when nodes are idle. This explains why energy consumption seems to be lower (due to improved performance) even when the SWP duty cycle is higher.

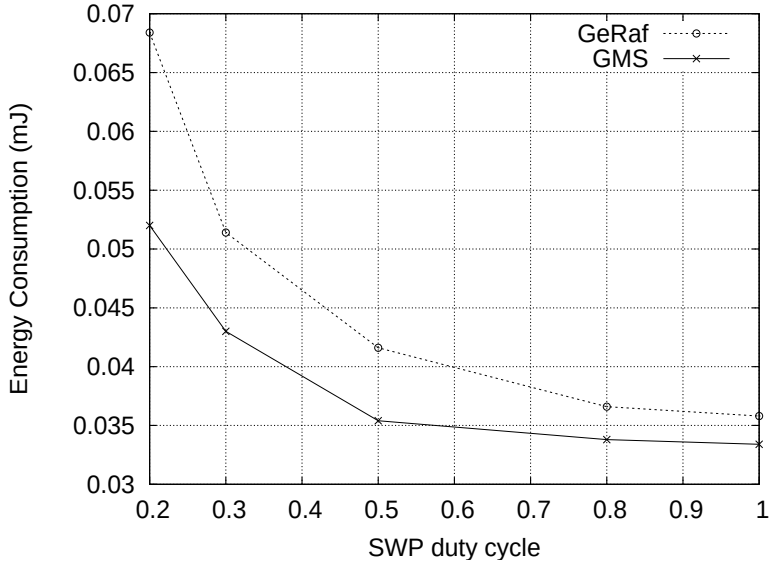
5.5.7 Discussion

GPSR always has a higher energy consumption. This is because of the extra beacon packets exchanged periodically. Due to the beacon packets, there is a trade-off between the energy consumption and the accuracy of the routing information. It is possible that by increasing the rate of beacon packets, the delay can be shortened, with the price paid in terms of higher energy consumption. We did investigate this issue, but found that by increasing the beacon rate of GPSR, the energy consumption greatly increased, without much improvement in the delay performance. This may be due to the fact that the beacon traffic is an additional traffic burden on the network and, that tends to increase delay.

Results show that GMS is better than GeRaF. The reasons are: GMS uses the LPN and geographic-Sift which helps to minimize the CTS collision and, at the same time, find best available relay node within short time.



(a) Delay versus SWP duty cycle.



(b) Energy consumption versus SWP duty cycle.

Figure 5.5: Case 3 (with fading & SWP): delay, energy consumption and packet loss rate.

GeRaF uses a more complicated way to determine the relay node. In GeRaF, CTS packet collisions occurs more frequently and the relay node can only be found after collisions have been resolved.

5.6 Conclusion

In this chapter, we proposed a novel hybrid opportunistic packet forwarding protocol for wireless sensor networks which we refer to as the *Geographic Multi-hop Sift* (GMS) protocol. The important feature of GMS is that it seamlessly combines the LPN, which is specified by the sender and to whom the highest priority is given, with the geographic Sift distribution. By doing this, it improves the efficiency whilst being robust to link or node failures. In addition, it is able to overcome the problems encountered by other similar schemes such as high probability of packet collisions and periodic information exchanging. The geographic Sift distribution uses the location information to determine a distribution. It gives preference to nodes with more forwarding progress, minimizes the probability of collision, while being robust to stochastic variations in the number of nodes that can hear the transmission.

We showed that the GMS protocol works well in a wide range of node densities, fading conditions and sleep-wake duty cycles, and is able to achieve better performance compared to GeRaF and GPSR. The cooperative nature of the process that determines the next hop node is an advantage for GMS when neighbors go to sleep often, or channel condition is bad. GMS adapts to this situation without requiring state information to be held in the nodes

themselves.

Finally, the performance of the GMS protocol is amenable to analysis which shows that it is optimal in terms of maximizing the rewards related to the progress being made towards the destination with each transmission.

Chapter 6

Generic Priority List Cooperation

From all the previous chapters, it is clear that coordination plays an important role in opportunistic cooperation. One of (if not) the most commonly used coordination scheme is priority list. In this chapter, we would like to propose a generic scheme of priority list coordination. Based on this scheme, we provide an algorithm which can find the optimal list, i.e. maximize the expected utility while minimizing the expected cost.

6.1 Introduction

In many opportunistic cooperation scenarios, it requires the cooperation seeker to find a good partner to cooperate with. In such cases, one of the most important tasks for the seeker is to select the partner among a set of candidates. However, as has been discussed before, the main difficulty

is that, we may not know which node is good and, more importantly, is available before the selection process. Hence, a smart coordination scheme is the key to the selection process, which is also the key to an opportunistic cooperation process.

As introduced in chapter 5, one of the widely implemented coordination techniques is the priority list. Before the coordination process, a group of nodes are selected as candidates; and during the coordination process, priorities are assigned to each candidate node. Based on the given priority, candidates sequentially send back their response to show their availability. One of the common techniques, which is adopted in many priority-list techniques, is to allow the coordination initiator to terminate the process. It means, when the initiator thinks that a good cooperation node has been found, it can terminate the node selection process to save energy and time. This normally happens when the first response message was heard by the initiator. With this technique, the priority assignment becomes more important, because nodes with low priority will have little chance of being selected.

After surveying many of the existing priority list techniques, we find out one common pattern of implementing the technique. That is the priorities are assigned only to find the best available cooperation partner, while neglecting the cost incurred during the coordination process. It means, the higher priorities are always given to nodes with better average performance (benefits). However, as we discussed before, the opportunistic cooperation may induce a substantial amount of overhead. It will be meaningless to get

the best available candidate, if a huge cost need to be paid before hand. Therefore, we think it is more reasonable to take both the expected benefits and the cost of coordination into consideration. For example, although some nodes may bring very good performance, if most of time they are not available, it may not be a good choice to assign high priorities to them.

In this chapter, we generalize this idea to a new metric, which is called cost aware utility (CAU). As the name CAU suggested, it takes into consideration both the benefit and cost. To apply the CAU into priority list coordination, we provide a genetic model to measure the performance of each node in the priority list. Then, we give a close form expression of the CAU value if a priority list is given. More importantly, based on this model, algorithm of finding the optimal priority assignment, given a list; and algorithm of finding the optimal priority list, given a node/candidate pool, are provided. These two algorithms have polynomial complexity and are proved to find the exact optimal solution.

6.2 Problem Formulation

6.2.1 Cost aware utility

To measure the performance of an Opportunistic Cooperation protocol, we propose a new metric, which is called cost aware utility (CAU). We define π as the coordination mechanism; B as the benefit; C as the coordination cost. Then suppose at time t, node i is selected as the relay node with expected benefit of $B_\pi(t)$. The cost of the coordination is $C_\pi(t)$. Then the CAU of this coordination mechanism is: $CAU(\pi) = \lim_{t\to\infty} \frac{\sum_{t=1}^{T}(B_\pi(t)-C_\pi(t))}{T}$. We

assume this performance is ergodic, then $CAU(\pi) = E(B_\pi - C_\pi)$. Both the B and C are generic terms, as long as they are matchable. For example, if the delay is the main concern, we can define B as the time saving gain, and define C as the time taken by the coordination process. Furthermore, the B and C can also represent a linear combination of a few metrics.

6.2.2 The priority list

For each cooperation candidate i, we use four variables, (B_i, FC_i, OC_i, PF_i), to characterized them. We assume that all these variables are known or can be estimated before the real coordination.

- B_i represents the expected benefit, to be received, if node i is selected as the partner.
- FC_i represents the fixed cost induced, if node i is enrolled in the priority list. In most cases, it means the extra cost of broadcasting the ID of node i in the cooperation request.
- OC_i represents the opportunistic cost induced, if node i is given a chance to send the response. It means, the extra energy and delay node i caused as it sends back the response message. Hence, if the coordination process ends before node i sends back the response, the OC_i should not be counted in the total cost.
- PF_i represents the probability that the cooperation initiator fails to receive the response from node i, after a chance is given to node i. The reasons of this failure may be due to node i being busy, or a bad

channel between the sender and node i.

Once, the value of these four variables are known for all the candidates, the CAU value can be determined for any given priority list. For example, if the priority list contains just one node, e.g. node A, $CAU(A) = B_A(1 - PF_A) - OC_A - FC_A$. However, the expression of the CAU, for a list with multiple nodes, is much more completed. In order to make the expression more clearer, let us define a new term, the utility without fixed cost: U. Clearly, for the one node list case, $U_A = B_A(1-PF_A)-OC_A$, and $CAU(A) = U_A - FC_A$.

A general priority list, with n nodes, L_n can be defined as: $< A1, A2, \ldots, An >$, and the list $L_{L_n-A1} =< A2, A3, \ldots, An >$. Then, the U for a general list can be shown below:

$$\begin{aligned} U(L_n) &= B_{A1}(1 - PF_{A1}) - OC_{A1} + PF_{A1}(U(L_{L_n-A1}) \\ &= U_{A1} + PF_{A1}(U(L_{L_n-A1}) \end{aligned} \tag{6.1}$$

Clearly, the U can be decomposed by two parts, the first node and the list after the first node. If the four attributes for all nodes are known, we can calculate the U by a recursive decomposition until the end of the list. Once, we have the $U(L_N)$, then, $CAU(L_n)$ is:

$$CAU(L_n) = U(L_n) - FC_{A1} - FC_{A2} \ldots - FC_{An} \tag{6.2}$$

6.3 Creating the optimal priority list

We believe that, the CAU can represent the generic performance objective of opportunistic cooperation protocol. Hence, our objective for this part is to find an algorithm to create the priority list, which has the maximum CAU value. To tackle this problem, we are going to separate it into two smaller problems. First problem, given a set of nodes, with the requirement that all the nodes are to be put into the priority list, what is the priority to be assigned to each node? We call this the optimal sequence problem. Once, this is solved, it means we can easily find out the optimal list with the constraint of all-node-in for a set of node. Second problem, given a set of nodes, how to find out a subset of these nodes, so that the optimal list for this subset, with the all-node-in constraint, are the optimal among all the possible subset. We call this the optimal subset problem. If both of the sub problems are solved, the original problem are solved.

6.3.1 The optimal sequence problem

Since each node can represent a one node list, it has its own CAU value. Intuitively, nodes with higher CAU should be assigned higher priorities. However, we find that, such an assignment is not always optimal. Suppose we have two candidates, $A1$ and $A2$. Let $B_{A1} = 10$, $FC_{A1} = 1$, $OC_{A1} = 2$, $PF_{A1} = 0.2$ and $B_{A2} = 15$, $FC_{A2} = 1$, $OC_{A2} = 2$, $PF_{A2} = 0.5$, then $CAU(A1) = 5 > CAU(A2) = 4.5$. However, comparing the list of $(A1, A2)$ with the list $(A2, A1)$, we find $CAU(A1, A2) = 5.1 < CAU(A2, A1) = 6.5$. Clearly it is better to assign node $A2$ with a higher priority, in this case.

In order to find the optimal sequence of a set of nodes, we define a new metric: sequence index (S: $S(\cdot) = B(\cdot) - \frac{OC(\cdot)}{1-PF(\cdot)}$) for each node. We find that the optimal sequence of a priority list should follows the order of the sequence index. This can be shown by the following theorems:

Theorem 6.1 *The relative priority between two nodes: for any two arbitrary nodes A1 and A2, if* $S_{A1} \geq S_{A2}$*, then* $CAU(A1, A2) \geq CAU(A2, A1)$

Proof: By the given condition, we have:

$$\begin{aligned} B_{A1} - \frac{OC_{A1}}{1-PF_{A1}} &\geq B_{A2} - \frac{OC_{A2}}{1-PF_{A2}} \\ \Rightarrow \frac{U_{A1}}{1-PF_{A1}} &\geq \frac{U_{A2}}{1-PF_{A2}} \end{aligned} \tag{6.3}$$

Since, the $U(\cdot) \geq 0$ (otherwise, this node should not be selected as a candidate) and $1 - PF(\cdot) \geq 0$, we have:

$$\begin{aligned} U_{A1}(1-PF_{A2}) &\geq U_{A2}(1-PF_{A1}) \\ \Rightarrow U_{A1} + PF_{A1}U_{A2} &\geq U_{A2} + PF_{A2}U_{A1} \\ \Rightarrow U_{A1} + PF_{A1}U_{A2} - CF_{A1} - CF_{A2} &\geq U_{A2} + PF_{A2}U_{A1} - CF_{A1} - CF_{A2} \\ \Rightarrow CAU(A1, A2) &\geq CAU(A2, A1) \end{aligned} \tag{6.4}$$

∎

Theorem 6.2 *The isolation property: we assume: 1)* $L1$ *and* $L2$ *are two arbitrary priority lists with mutually exclusive nodes; 2)* $A1$ *and* $A2$ *are two arbitrary nodes, both of which are not in the list of* $L1$ *or* $L2$*; 3)* $CAU(A1, A2) \geq CAU(A2, A1)$*. With these conditions, the statement:*

$CAU(L1, A1, A2, L2) \geq CAU(L1, A2, A1, L2)$ *is always true.*

Proof: It is obvious that, if $CAU(A1, A2, L2) \geq CAU(A2, A1, L2)$, then $CAU(L1, A1, A2, L2) \geq CAU(L1, A2, A1, L2)$. Hence, we want to prove that $CAU(A1, A2, L2) \geq CAU(A2, A1, L2)$ is true with given condition.

Since we assume $CAU(A1, A2) \geq CAU(A2, A1)$, we have:

$$U_{A1} + PF_{A1}U_{A2} \geq U_{A2} + PF_{A2}U_{A1} \tag{6.5}$$

By defining the term $U(L2)$ and FC_{L2} as the utility without fixed cost of list $L2$ and the sum of fixed cost of all the nodes in the list $L2$ respectively, we have:

$$\begin{aligned} U(A1, A2, L2) &= U_{A1} + PF_{A1}U_{A2} + PF_{A1}PF_{A2}U(L2) \\ &\geq U_{A2} + PF_{A2}U_{A1} + PF_{A2}PF_{A1}U(L2) \\ &= U(A2, A1, L2) \end{aligned}$$

Since $U(A1, A2, L2) \geq U(A2, A1, L2)$, then $CAU(A1, A2, L2) \geq CAU(A2, A1, L2)$

∎

With these two theorems, we can easily find the optimal sequence for any set of relay candidates. This can be shown in the following lemma:

Lemma 6.1 *Given an arbitrary set of n relay candidates, let S_{Ai} represents the sequence index of node A_i, and assumes $S_{A1} \geq S_{A2} \geq S_{A3} \geq \cdots S_{An}$; then $CAU(A_1, A_2, A_3, \cdots A_n) \geq CAU(x_n)$, where (x_n) represent any list formed by these n nodes.*

6.3.2 The optimal subset problem

We have solved the sequence problem. However, this is just half of the optimal priority list problem. With lemma 6.1 we can only find out the optimal sequence, hence the optimal list, if all nodes have to be selected in the list. However, as discussed before, putting all nodes into the list may not be optimal. Hence, there is another more challenging problem: how to select a few relay candidates from all the given nodes. It means, giving a universal set of M nodes, we need to find the optimal subset, among all the possible subsets, which can give the highest CAU value, based on the sequence given by lemma 6.1.

This problem can be easily formulated as an integer programming problem. However, to find the optimal list, the computational complexity by this formulation increases exponentially with the increasing of number of nodes M. Hence, this formulation is not practically sound.

We have discovered that, if we can assume that the fixed cost (FC) for all the nodes are either negligible or equal to each other, then this problem can be solved in polynomial time. Since the FC for normal application, is just the power consumption and delay caused when sender broadcasts the priority list. We believe that our assumption holds for most of the common network applications.

Given a set of M nodes, let OU represent the optimal subset and OU_1, $OU_2 \cdots$ represent the optimal one-node subset, the optimal two-node subset, etc. The optimal t-node subset just means the subset, which can give the highest CAU value among all possible subsets with exactly t nodes. Then

we have the following theorem.

Theorem 6.3 *Given a set of M arbitrary nodes, if the FCs for all nodes are either negligible or equal to each other, then $OU_1 \subset OU_2 \cdots \subset OU_M$.*

Proof: If the CFs for all nodes are equal, the selection of nodes in OU_i $(\forall i)$ does not depend on the value of CF. Hence, we can assign any value to CF which does not change the result. For simplicity, we let $CF = 0$. Under this case, for any node A: $CAU(A) = U_A$. In addition, breaking any list L into two parts L_1 (first part) L_2 (second part), at any position, we have:

$$U(L) = U(L_1) + P_{L1}U(L_2) \tag{6.6}$$

where P_{L1} is the joint product of PF for all nodes in L_1.

We use mathematical induction to prove the above statement. Firstly, It is easy to show that $OU_1 \subset OU_2$. Since OU_1 contains the node with highest utility U. If this node is not included in OU_2, we can always replace the second node in OU_2 with the node in OU_1 and get a higher, at least equal, CAU value. Secondly, we assume that, $OU_1 \subset OU_2 \cdots \subset OU_k$, $(k < M)$ is true. Lastly, we want to prove $OU_k \subset OU_{k+1}$

Let $A_1, A_2, \cdots, A_k$ represent the nodes in the list OU_k and $O\bar{U}_k$ represent the union of the rest $(M-k)$ nodes, which are not selected in OU_k. Then, it is clear that one-node list (A_k) is the best one-node list, with highest U, in the union $O\bar{U}_k \cup A_k$. Similarly, two-node list (A_{k-1}, A_k) is the best two-node list in the union $O\bar{U}_k \cup A_{k-1} \cup A_k$. In general, the best $(k-j)$ node list from

the union $A_(j+1), A_(j+2) \cdots A_k \cup \bar{OU}_k$ is $(A_(j+1), A_(j+2) \cdots A_k)$.

Assume a $k+1$ node list, OU'_{k+1}, that has more than 1 node being different from OU_k. Nodes selected by OU'_{k+1} are represented as $B_1, B_2, \cdots, B_{k+1}$, where the first different node is at B_{j+1}, and, j is any non-negative integer number. Hence, based on Lemma 6.1, node 1 to j are exactly the same between OU_k and OU'_{k+1}. Now, we want to show that this OU'_{k+1} is not optimal.

After node B_{j+1}, there are another $(k-j)$ nodes, which can be selected from the rest of $M-j-1$ nodes. From Equation 6.6, we know that the key to the second part of the list is just the U value. We know, by OU_k, the best $(k-j)$ node list from $M-j$ nodes are $A_{j+1}, A_{j+2} \cdots A_k$. We also know that the first j nodes which been excluded are the same for both series A and B. Here, there is an extra node, the node B_{j+1}. This node is not selected in the series A. Hence, the list $A_{j+1}, A_{j+2} \cdots A_k$ is also the best $(k-j)$ node list, from the set of $M-j-1$ nodes. It means, if we put them after node B_{j+1}, it will give a higher overall CAU. On the other hand, this list contains all the nodes from series A: A_1 to A_j in the first part and A_{j+1} to A_k in the second part. Hence, it contains only one new node, which is contradictory to the assumption. ∎

Clearly, the optimal subset, OU, among the M nodes must be one of the OU_i, $(1 \leq i \leq M)$. Hence, we can get the OU by finding out the best list among all the OU_i. Based on this idea, we elaborate the algorithm we designed in Algorithm 5.

This algorithm has the complexity of M^3, where M is the total number

Algorithm 5: The algorithm to find the optimal priority list

Suppose there are M potential relay nodes with known value of PF, OC and B
Let the OU_0 be an empty list
forall M *nodes* **do**
  Calculate the value of utility, U, and sequence index, S,
end
Sort the sequence of nodes based on their S
for $i = 1$ **to** M **do**
  forall $M - i - 1$ *nodes, which is not included in* OU_{i-1} **do**
    Add itself into the list OU_{i-1}, base on the calculated sequence
    Calculate the temporary CAU of the temporary list
  end
  Find out the node A_j which can give the highest CAU value.
  Create OU_i by add node A_j into list OU_{i-1}
  Keep the CAU of the OU_i list
end
The final OU is the OU_i, which can give the highest CAU value among all the OU_is

of nodes. Most importantly, we can find the exact optimal priority list based on this algorithm for most common network applications, where the FCs are insignificant or have similar values for all the nodes.

6.4 Application and analysis of the algorithm on an opportunistic forwarding problem

In this section, we will show an example of how to apply this cost-aware coordination to a familiar networking problem, which is the opportunistic forwarding problem shown in the last chapter. Firstly, we will introduce the network structure, including the network topology, the related parameters and the assumptions. Then we will apply the newly proposed algorithm to solve it. Meanwhile, algorithm based on AnyCast [7] is also implemented

Transmission Rate	54 Mbps
RTS, CTS, ACK	352 bits
Trans Range (error free)	30 m
Trans Range (max)	60 m
Slot Time	20 μs
DIFS Duration	30 μs
SIFS Duration	10 μs
Avg Contention Time	40 μs

Table 6.1: The related network parameters.

for comparison. Finally, we will compare the difference of the expected performance between both of the algorithms by analysis.

6.4.1 The network structure

This is an IEEE 802.11g network. There are many wireless stations located in a line topology. The source node and destination nodes are located at the end of the line and all the relay nodes are uniformly distributed on this line. We assume that the error free transmission distance is 30 m, which means the transmission is always successful if the distance between the sender and receiver is less than 30 m. Beyond this range, the packet loss probability will gradually increase. The maximum transmission range is 60 m. As shown in Table 6.4.1, the set of core parameters used was assumed to be the default values of the IEEE 802.11g network.

In this problem, the main performance goal is to deliver the packet to the destination as soon as possible. Hence, we have to give the sender a good priority list in each step, with which it can transmit the data as far as possible and with as little overhead (time) as possible.

6.4.2 Modeling as a cost-aware opportunistic cooperation problem

In order to solve the problem using the cost-aware model, we need to define the tuple of $< B_i, FC_i, OC_i, PF_i >$ for each node i. The benefit of each node can be measured by the distance gain, the overhead (cost) can be measured by the extra delay induced, and the probability of packet loss can be measured in the networks. However, the main difficulty is that, the benefit and the cost have different units: one is the unit of distance and the other is the unit of time. Hence, we have to unify them before looking for the optimal priority list.

Considering the final objective is to minimize the overall transmission time, we want to convert the distance gain into time gain. To do so, we consider the node with the distance gain of 30 M as the reference point. The one hop transmission time of one of this relay node is shown in Equation (6.7),

$$\begin{aligned} T_{30} &= \frac{(L + 3 * 352)bits}{54Mbps} + 3T_{Sifs} + T_{Difs} + T_{cont} \\ &= \frac{(L + 1056}{54 * 1024^2} + 100\mu s \end{aligned} \tag{6.7}$$

where L is the packet size, T_{Sifs}, T_{Difs}, and T_{cont} are the time of SIFS, DIFS and average contention, respectively.

Compared with the 30 m reference point, any node with a distance gain of d m, can be considered to have a time gain: $\frac{d-30}{30}T_{30}$, because the actual transmission time is the same for nodes with different distance gain (The

overhead of opportunistic cooperation is considered later). A node with d m distance gain can be considered to be $\frac{d}{30}$ hop of transmission.

Hence, the four variable of each node is:

- B: $\frac{d-30}{30}T_{30}$ for node with distance gain d
- OC: the additional time slot required, which is $20\mu s$ for all nodes
- FC: negligible
- PF: To be measured by the network

6.4.3 Analysis of the performance

In this analysis, we let the end-to-end distance between the source and destination equal 1000 m. The distance between each two consecutive nodes is 3 m. The performance is compared with two different protocols. One is a deterministic forwarding protocol, which always sends data to a node with 30 m distance gain. The other is an opportunistic forwarding protocol, based on AnyCast, which always gives higher priorities to nodes with higher distance gain and include all nodes (with the distance gain from 30- 60 m) in the priority list.

Case one: With different degree of packet loss increasing rate

According to many performance studies of IEEE 802.11 networks, the throughput performance is almost constant for a close range transmission. After a certain threshold, the throughput performance will gradually decrease. According to this study, we would like to model the packet loss rate as a similar

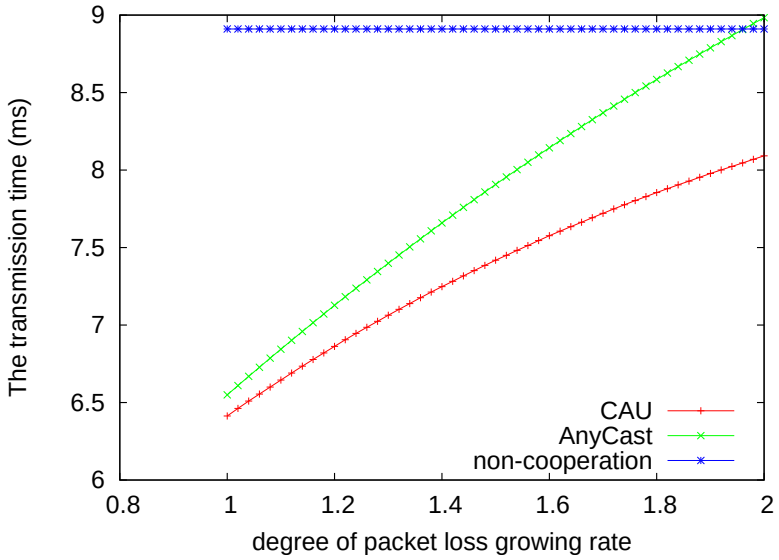


Figure 6.1: The average transmission time with different degree of the packet loss increasing rate.

function as the throughput. It means that in a close distance ($< 30M$), the loss rate is 0. Then, the packet loss rate will gradually increase. Beyond the maximum transmission range, the loss rate is 1. Mathematically: the PF is: $(\frac{d-30}{60-30})^x$, where $x(x \geq 1)$ is the degree of packet loss increase rate with the reference of the distance. It means, if $x = 1$ the packet loss rate increases linearly. With the increasing value of x, the nodes with a higher distance gain will have a higher packet loss probability. However, for nodes with little distance gain, they will have less packet loss probabilities. By fixing the packet size at 1 KB, we compare the performance among the three algorithms with different x.

The performance of the three different protocols is shown in Figure 6.1. Clearly, the non-cooperative protocol has a constant transmission time, because it always chooses the node with 30 m distance gain and 0 packet loss

rate. Our protocol, which considers the cost-aware utility, has the lowest transmission time (delay). This shows how well the protocol can adapt to different channel conditions. Lastly, the results show that, the AnyCast protocol performs well when the loss rate is close to linear ($x = 1$). However, when x is close to 2, the AnyCast protocol performs even worse than the non-cooperative protocol. This is because, when x is higher, nodes with better distance gains have much worse channel condition than before. However, AnyCast protocol will not take this into consideration but keeps on giving high priority to these nodes. As a result, most of the time, the first few slot are wasted, which leads to higher cost.

Case two: Performance difference with different packet size

In the second case, x is fixed at 1 but the packet size is variable. By changing the packet size from 512 Bytes to 1024 Bytes, we would like to see the differences in performance between the three protocols.

As shown in Figure 6.2, the performance of the three protocols has a similar pattern. The optimal CAU performs better than the AnyCast, and both of them are better than the non-cooperation one. The performance of the three protocols under different packet sizes is similar. When the data size is small, the average transmission time is shorter for all three protocols, and vice versa. However, there is an interesting point shown in the graph: when the packet size is small, the difference between optimal CAU and AnyCast is large; however, when the packet size is small the performance between optimal CAU and the non-cooperative protocol is small. The reason is that, when the packet size is small, the overhead is relatively large. Hence, an

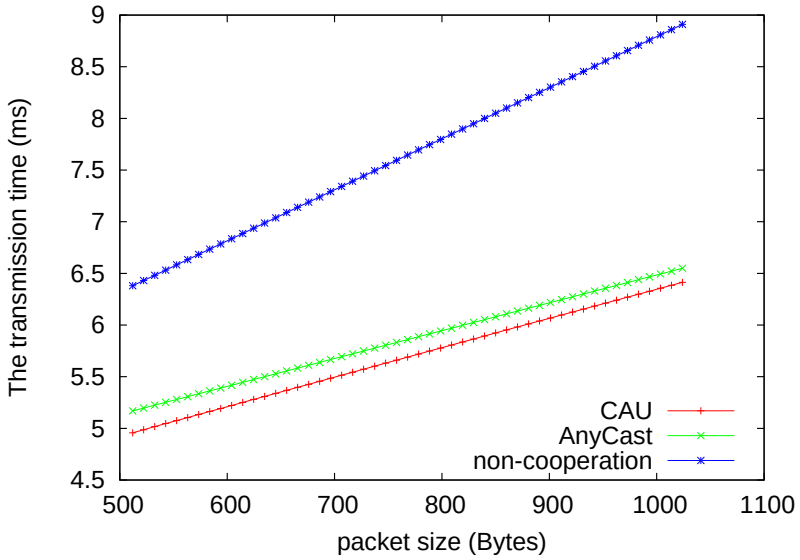


Figure 6.2: The average transmission time with different packet size.

opportunistic cooperation protocol becomes less beneficial. That is why, compared with the non-cooperative protocol, both the optimal CAU and the AnyCast has less gains when the packet size is small. However, between the optimal CAU and the AnyCast, the optimal CAU is more cost aware, while AnyCast is less. Hence, optimal CAU performs better than AnyCast when data size is small.

6.5 Conclusion

In this chapter, we propose a generic model for priority list based coordination techniques, which models each node by four variables and takes both the overhead and potential benefits into consideration. Based on this scheme, we design an algorithm to find the optimal sequence assignment among a given set of nodes. Furthermore, when the fixed cost is similar among all

candidates, we propose an algorithm, of polynomial time complexity, to find the best priority list, which can give optimal expected performance before the real data transmission. By comparing the performance of the proposed algorithm with existing algorithms, like AnyCast and non-opportunistic cooperation protocols, we have verified that the proposed algorithm gives better performance than other protocols.

Chapter 7

Conclusion and Open Issues

Cooperation provides performance improvements through the use of available resources from multiple agents in the network. However, due to the dynamics of the networks or the lack of information of the networks, most of the time, there is only some unreliable opportunity of cooperation available. It means that, it is not certain whether cooperation will bring benefits and even if so, whom the cooperation should be performed with. Hence, it is interesting and meaningful to study opportunistic cooperation in real life, especially about the issues of information acquisition with online decision making, and related coordination schemes for opportunistic cooperation.

In this thesis, we studied these issues through two main applications. Firstly, we explored the benefits of cooperation and concurrent transmissions at the medium access control (MAC) layer in wireless LANs. We proposed two novel MAC (CCMAC and SI-CCMAC) protocols which utilizes these features to improve the throughput performance of the network. They take three steps before transmission: rate detection, helper selection and

packet shaping. Both protocols have different transmission modes. One of the modes is chosen based on the channel condition and the helper's status. The protocols enable up to 5 concurrent transmissions and can achieve substantial throughput performance improvement over the legacy IEEE 802.11, without incurring significant network overheads. Hence, we believe that they are good extensions of the existing WiFi MAC protocol.

In the second application, we proposed a novel hybrid opportunistic packet forwarding protocol for wireless sensor networks which we refer to as the *Geographic Multi-hop Sift* (GMS) protocol. The important feature of GMS is that it seamlessly combines the LPN, which is specified by the sender and to whom the highest priority is given, with the geographic Sift distribution. By doing this, it improves the efficiency whilst being robust to link or node failures. In addition, it is able to overcome the problems encountered by other similar schemes such as high probability of packet collisions and periodic information exchange. We showed that the GMS protocol works consistently well in a wide range of node densities, fading conditions and sleep-wake duty cycles, and is able to achieve better performance compared with GeRaF and GPSR. The cooperative nature of the process that determines the next hop node is an advantage for GMS when neighbors of the sender frequently go to sleep, or the channel condition is bad. GMS adapts to this situation without requiring state information to be held by the nodes themselves.

Lastly, we proposed a general coordination scheme for opportunistic cooperation. It is one of the priority list technique. One important feature of

this scheme is that, it takes both the expected future benefits and the overhead incurred by the coordination into consideration and unifies them into a single metric CAU (cost aware utility). Based on this scheme, an algorithm has been proposed to find the optimal sequence among a given candidate list. Furthermore, algorithm for the common case, where the fixed cost of all the nodes are similar, is also given, which can find the optimal priority list among a set of candidates.

7.1 Open Issues

In the area of opportunistic cooperation, there are still many unsolved issues. One of them is the security issue. A misbehaving partner can degrade the envisaged performance improvements severely. In practice, there are no mechanisms to ensure adherence of the partner to the cooperation strategy. The common way to tackle this challenge is to identify and isolate the misbehaving nodes during the transmission. Researchers have done a lot of studies and proposed various mechanisms to tackle this challenge, such as [63], [64], [65], and etc. However, most of them need long histories of data and heavy communication between peers, in order to find the misbehaving nodes. This makes them unsuitable to be applied in opportunistic cooperation scenarios. Hence, one of the open problems is to design a light weight, low overhead mechanism to detect the misbehaving nodes.

Another open issue is about the distributed coordination in opportunistic cooperation. To save energy and reduce overhead, we may choose a distributed way of coordination between all the candidates. This is another

step forward from the decision making with incomplete knowledge. However, it is more challenging in the sense that, it may need more nodes to make judgment based on their own knowledge of the network. Distributed learning algorithms may be a suitable solution to be applied in these cases. However, how to reduce the computational complexity and convergence time will be another issue in these case.

In summary, in this thesis we investigated some undeveloped area in wireless networks, where opportunistic cooperation can be suitably =applied. We obtained some interesting results and learned a lot of lessons during this process. We hope our work can help future researchers to have a deep understanding of this area and inspire them to further investigate this field.

Bibliography

[1] J. N. Laneman, D. N. C. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: efficient protocols and outage behavior," *IEEE Trans. Inform. Theory*, vol. 50, pp. 3062–3080, 2004.

[2] T. M. Cover and A. A. E. Gamal, "Capacity theorems for the relay channel," *IEEE Trans. Inform. Theory*, vol. 25, p. 572584, 1979.

[3] W. hin Wong, J. M. Shea, and T. F. Wong, "Cooperative-diversity slotted ALOHA," *Wireless networks*, vol. 13, p. 361369, 2007.

[4] C.-T. Chou, J. Yang, and D. Wang, "Cooperative mac protocol with automatic relay selection in distributed wireless networks," in *PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, 2007.

[5] P. Liu, Z. Tao, and et. al., "CoopMAC: A Cooperative MAC for Wireless LANs," *IEEE JSAC*, vol. 25, no. 2, pp. 340–354, 2007.

[6] H. Zhu and G. Cao, "rDCF: A relay-enabled medium access control protocol for wireless ad hoc networks," in *Proc. of INFOCOM*, 2005.

[7] S. Jain and S. Das, "Exploiting path diversity in the link layer in wireless ad hoc networks," *IEEE WoWMoM symposium*, June 2005.

[8] S. Biswas and R. Morris, "Opportunistic routing in multi-hop wireless networks," in *Proceedings of the IEEE/ACM SIGCOMM*, 2005.

[9] P. Larsson, "Selection diversity forwarding in a multihop packet radio network with fading channel and capture," *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 4, pp. 47–54, 2001.

[10] R. R. Choudhury and N. H. Vaidya, "MAC-layer anycasting in ad hoc networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, pp. 75–80, 2004.

[11] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and latency performance," *IEEE Transactions on Mobile Computing*, vol. 02, pp. 349–365, Oct-Dec 2003.

[12] M. Witt and V. Turau, "BGR: blind geographic routing for sensor networks," in *Third International Workshop on Intelligent Solutions in Embedded Systems*, pp. 51–61, May 2005.

[13] Y. Yuan, H. Yang, S. Wong, S. Lu, and W. Arbaugh, "ROMER: Resilient Opportunistic Mesh Routing for Wireless Mesh Networks," in *First IEEE Workshop on Wireless Mesh Networks*, September 2005.

[14] R. Bellman, "A Markovian Decision Process," *Journal of Mathematics and Mechanics*, vol. 6.

[15] R. A. Howard, "Dynamic programming and markov processes.," 1960.

[16] M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some simplified np-complete problems," in *STOC '74: Proceedings of the sixth annual ACM symposium on Theory of computing*, pp. 47–63, ACM, 1974.

[17] C. C. Wang, "An algorithm for the chromatic number of a graph," *J. ACM*, vol. 21, no. 3, pp. 385–391, 1974.

[18] S. Skiena, "Finding a vertex coloring.," *5.5.3 in Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica. Reading*, pp. 214–215, 1990.

[19] A. M. C. Lucet, F. Mendes, "An exact method for graph coloring.," *Computers and Operations Research*, no. 8, pp. 2189–2207, 2006.

[20] G. Chaitin, "Register allocation and spilling via graph coloring," *SIGPLAN Not.*, vol. 39, no. 4, pp. 66–74, 2004.

[21] H. A. Peelle, "Graph coloring in j: an introduction," in *APL '01: Proceedings of the 2001 conference on APL*, pp. 77–82, ACM, 2001.

[22] M. Caramia and P. Dell'Olmo, "Coloring graphs by iterated local search traversing feasible and infeasible solutions," *Discrete Appl. Math.*, vol. 156, no. 2, pp. 201–217, 2008.

[23] Gendron, Bernard, Hertz, Alain, St-Louis, and Patrick, "On edge orienting methods for graph coloring," *Journal of Combinatorial Optimization*, vol. 13, pp. 163–178, February 2007.

[24] P. Galinier, A. Hertz, and N. Zufferey, "An adaptive memory algorithm for the k-coloring problem," *Discrete Appl. Math.*, vol. 156, no. 2, pp. 267–279, 2008.

[25] C. A. Glass and A. P. Bennett, "A polynomially searchable exponential neighbourhood for graph colouring," *Journal of the Operational Research Society*, vol. 56, no. 3, 2005.

[26] H. S. Wilf, "Algorithms and Complexity," (London, UK.), Prentice Hall Inc, 1986.

[27] R. Beigel, "Finding maximum independent sets in sparse and general graphs," in *SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, (Philadelphia, PA, USA), pp. 856–857, Society for Industrial and Applied Mathematics, 1999.

[28] A. Grosso, M. Locatelli, and W. Pullan, "Simple ingredients leading to very efficient heuristics for the maximum clique problem," *Journal of Heuristics*, vol. 14, no. 6, pp. 587–612, 2008.

[29] Status, "of Project IEEE 802.11n, IEEE 802.11 - task group n - meeting update," *http://grouper.ieee.org/groups/802/11/Reports.*

[30] M. Gastpar, G. Kramer, and P. Gupta, "The multiple-relay channel: coding and antenna-clustering capacity," *In Proceedings of IEEE Information Theory*, 2002.

[31] IEEE, "802.11 standard," *http://standards.ieee.org/getieee802/802.11.html.*

[32] P. Karn, "MACA – a new channel access method for packet radio," in *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pp. 134–140, September 1990.

[33] V. Bharghavan and et. al, "MACAW: A media access protocol for wireless LAN's," in *ACM SIGCOMM*, pp. 212–225, 1994.

[34] A. Kamerman and L. Monteban, "WaveLAN-II: a high-performance wireless LAN for the unlicensed band," *Bell Labs Technical Journal*, vol. 2, no. 3, 1997.

[35] G. Holland, N. Vaidya, and P. Bahl, "A rate-adaptive MAC protocol for multi-hop wireless networks," in *Proceedings of the ACM international conference on Mobile computing and networking*, 2001.

[36] H. Zhu and G. Cao, "On improving the performance of IEEE 802.11 with relay-enabled PCF," *Mobile Networks and Applications*, vol. 9, no. 4, 2004.

[37] S. Narayanan and S. S. Panwar, "To forward or not to forward - that is the question," *Wirel. Pers. Commun.*, vol. 43, no. 1, 2007.

[38] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," *MIT Press, Cambridge, MA, 1998.*

[39] D. A. Aberdeen, "Policy-Gradient Algorithms for Partially Observable Markov Decision Process," *Ph.D Thesis, National ICT Australia, Canberra, 2003.*

[40] B. Bonet, "An epsilon-optimal grid-based algorithm for partially observable markov decision processes," in *Proceedings of the International Conference on Machine Learning*, 2002.

[41] OmNet++Community, "http://www.omnetpp.org/,"

[42] M. Zorzi, R. R. Rao, and L. B. Milstein, "On the accuracy of a first-order Markov model for data transmission on fading channels," in *IEEE ICUPC'95*, November 1995.

[43] Q. Zhang and S. Kassam, "Finite-state Markov model for Rayleigh fading channels," *IEEE Trans. Commun*, vol. 47, no. 11, pp. 1688–1692, 1999.

[44] D. Stiliadis and A. Varma, "Latency-rate servers: a general model for analysis of traffic scheduling algorithms," *IEEE/ACM Trans. Netw.*, pp. 611–624, 1998.

[45] W. Wang, Y. Wang, X.-Y. Li, W.-Z. Song, and O. Frieder, "Efficient interference-aware tdma link scheduling for static wireless networks," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 262–273, ACM, 2006.

[46] B. Escoffier, J. Monnot, and V. T. Paschos, "Weighted coloring: further complexity and approximability results," *Inf. Process. Lett.*, vol. 97, no. 3, pp. 98–103, 2006.

[47] J. Xue, "Solving the minimum weighted integer coloring problem," *Comput. Optim. Appl.*, vol. 11, no. 1, pp. 53–64, 1998.

[48] D. Eppstein, "Small maximal independent sets and faster exact graph coloring," in *WADS '01: Proceedings of the 7th International Workshop on Algorithms and Data Structures*, (London, UK), pp. 462–470, Springer-Verlag, 2001.

[49] L. Tassiulas and A. Ephremides, "Stability Properties of Constrained Queueing Systems and Scheduling policies for Maximum Throughput in Multihop Radio Networks," *IEEE Trans. Automat Control*, vol. 37, no. 12, 1992.

[50] Z. Hu and C.-K. Tham, "CCMAC: Coordinated cooperative MAC for wireless LANs," in *MSWiM '08*, pp. 60–69, ACM, 2008.

[51] C.-S. Chow and J. N. Tsitsiklis, "The complexity of dynamic programming," *J. Complex.*, vol. 5, no. 4, pp. 466–488, 1989.

[52] C.-S. Chow and J. N. Tsitsiklis, "An optimal one-way multigrid algorithm for discrete-time stochastic control," *IEEE transactions on automatic control.*, vol. 36, pp. 898–914, 1991.

[53] Y. C. Tay, K. Jamieson, and H. Balakrishnan, "Collision-minimizing CSMA and its applications to wireless sensor networks," *IEEE JSAC*, vol. 22, pp. 1048–1057, August 2004.

[54] B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *MobiCom*, August 2000.

[55] "IEEE Computer Society LAN MAN Standards Committee. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specification."

[56] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 12, June 2004.

[57] D. B. Johnson and D. B. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, pp. 153–181, 1996.

[58] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing, IETF RFC 3561," July 2003.

[59] Y. Yu, R. Govindan, and D. Estrin, "Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks," *UCLA Computer Science Department Technical Report UCLA/CSD-TR-01-0023*, May 2001.

[60] M. Heissenbuttel, T. Braun, T. Bernoulli, and M. Walchli, "BLR: Beacon-Less Routing algorithm for mobile ad-hoc networks," *Computer Communications*, vol. 27, no. 11.

[61] B. Blum, T. He, S. Son, and J. Stankovic, "IGF: A state-free robust communication protocol for wireless sensor networks," tech. rep., Technical Report CS-2003-11, Department of Computer Science, University of Virginia, 2003.

[62] "Crossbow MICA2 documentation." Available online at: http://www.xbow.com.

[63] M. Gupta, P. Judge, and M. Ammar, "A reputation system for peer-to-peer networks," *NOSSDAV*, June 2003.

[64] L. Xiong and L. Liu, "Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 843–857, 2004.

[65] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Trans. Sen. Netw.*, vol. 4, no. 3, pp. 1–37, 2008.

List of Publications

- **Zhengqing Hu**, Chen-Khong Tham,*SI-CCMAC: Sender Initiated Concurrent Cooperative MAC for Wireless LANs*, in Proceedings of **WiOpt 2009**, Seoul, Korea
- **Zhengqing Hu**, Chen-Khong Tham,*CCMAC: Coordinated Cooperative MAC for wireless LANs* , in Proceedings of **ACM MSWiM 2008**, Vancouver, Canada
- **Zhengqing Hu**, Chen-Khong Tham,*HOF: Hybrid Opportunistic Forwarding for Multi-Hop Wireless Mesh Networks*, in Proceedings of **ICC 2008**, Beijing, China
- **Zhengqing Hu**, Chen-Khong Tham,*CCMAC: Coordinated Cooperative MAC for wireless LANs*, **Computer Networks** Volume 54, Issue 4, 19 March 2010, Pages 618-630
- **Zhengqing Hu**, Chen-Khong Tham,*GMS: Geographic Multi-hop Sift*, submitted to IEEE Transactions on Mobile Computing