CCMAC: Coordinated Cooperative MAC for Wireless LANs

Zhengqing Hu
Department of ECE
National University of Singapore
zhengqing@nus.edu.sg

Chen-Khong Tham
Department of ECE
National University of Singapore
eletck@nus.edu.sg

ABSTRACT

In wireless LANs, one of the main concerns is throughput performance. When there is only one Access Point (AP) in a wireless LAN, the bottleneck is normally at the region near the AP. In this paper, 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 low data-rate direct link. Furthermore, it can coordinate nodes to perform concurrent transmissions in order to further increase throughput. This coordination is done by modeling the problem as a POMDP (Partially Observable Markov Decision Process) and using a Reinforcement Learning (RL) algorithm to solve it. 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 it has better throughput performance than other MAC protocols, such as CoopMAC and legacy IEEE 802.11.

Categories and Subject Descriptors: C.2.1 [Network Architecture and Design]: Wireless communication

General Terms: Algorithms

Keywords: MAC, concurrent, cooperative

1. INTRODUCTION

IEEE 802.11 (WiFi) based wireless LANs have become extremely popular in the past decade. One of the main reasons for its 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 [14] supports data rates up to 248Mbps. However, in the real world, it may be more important to consider the effective data rate, which is the 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. Hence, one of the most important and practical problems in wireless network protocol design is to combat these negative effects in order to achieve a large overall throughput.

To mitigate some of the above mentioned problems, techniques called cooperative communications, such as [10], [19], [4], are being developed. Due to the broadcast nature of 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.

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. Our second observation is that, when cooperative communication is applied, concurrent transmissions are still possible, even if the transmissions are from different nodes to the same destination. 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 5, is that, by modeling the underlying problem as a Partially Observable Markov Decision Process (POMDP), we can use a Reinforcement Learning (RL) algorithm to coordinate the senders and optimize the throughput performance.

In this paper, we propose a novel coordinated cooperative MAC (CCMAC) protocol. It is designed for the uplinks[1] (from clients to AP) of the AP's one-hop region. CCMAC can intelligently apply cooperative transmission, by two-hop relaying, and coordinate up to five concurrent transmissions within this region. We evaluate the performance of CCMAC by analysis and simulation, and show that CCMAC can reduce the transmission time, and hence, increase the

MSWiM'08, October 27–31, 2008, Vancouver, BC, Canada.

---

[1] Similar to CCMAC and incorporating bulk transmission, a protocol for the downlink will be proposed in another paper.

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 [10].

Our main contributions in this paper are:

- the CCMAC protocol, which can intelligently apply two-hop transmission and enable concurrent transmissions for nodes with unfavorable channels to the AP;
- use of POMDP to model the coordination problem and application of an RL algorithm to solve it;
- analysis of the maximum number of concurrent transmission achievable in the one-hop region of AP; and
- evaluation of the performance of CCMAC by analysis and simulation.

2. IEEE 802.11 AND RELATED WORK

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. Hence, WiFi has the capability to choose the data rate adaptively.

There are two modes of the MAC protocol operation in WiFi. One is PCF (point coordination function), the other is DCF (distributed coordination function). Between them, DCF is more widely used. The standard DCF protocol is described in [6]. 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, since not all stations can hear each other, packet collision may still occur even if the channel is sensed clear. This is the well-known hidden terminal problem. To solve this problem, the RTS-CTS handshaking, which was first designed in MACA [9] and modified in MACAW [2], is also employed in WiFi. The sender sends an RTS packet and the receiver sends a CTS packet 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. The RTS-CTS handshaking can largely solve the hidden terminal problem. However, sending the RTS-CTS packets themselves causes additional overhead. In IEEE 802.11, the RTS-CTS mechanism is applied when the data packet is larger than a certain threshold.

2.2 Related Work

The ARF protocol [8] 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 RBAR protocol [5] 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 [18] proposed the 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 [19] and CoopMAC [10], were 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/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, this capability further increases the throughput performance in the near-AP region.

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. Improving the throughput performance in this region can improve the throughput performance for the whole network. CCMAC is specially designed to operate in this region. It employs two techniques: cooperative transmission and concurrent transmission. In this section, we explain why these techniques can help to increase the throughput performance.

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 1, suppose the data rate of direct transmission from the source node, S_{s1}, to the destination node, AP, is R_{sd}, which 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. source node sends data to helper node, then the helper node relays the data to the destination node, it may 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 the destination with the two-hop relaying is $\frac{1}{R_{sh}} + \frac{1}{R_{hd}}$. Hence, as long as equation 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.[2]

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

---

[2] For simplicity, transmission overhead is not considered here. However, it is considered in the protocol design and analysis part.

3.2 Advantages of concurrent transmission in Wireless LANs

In the uplink (traffic from user nodes to AP) of a wireless LAN, there may be multiple source nodes and multiple helper nodes but one destination node, which is the AP. In the AP's one-hop region, because of the single destination (AP), concurrent transmission is impossible when direct transmission is applied. However, when two-hop relaying is implemented, concurrent transmission becomes possible. The basic idea of the proposed concurrent transmission 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 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 AP one after another. By using concurrent transmission, 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 (it is proved in Theorem 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 transmission scheme is shown in equation 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 condition, the total transmission time T_{nonC} for the non-concurrent two-hop transmission is shown in equation 3. Clearly, $T_C = T_{nonC}$ when $n = 1$ and $T_C < T_{nonC}$ when $n > 1$. From these equations we can see that, two-hop concurrent transmission can achieve even more throughput than a two-hop non-concurrent transmission.

$$T_C = \max_i \frac{1}{R_{i,sh}} + \sum_{i=1}^{n} \frac{1}{R_{i,hd}} \tag{2}$$

$$T_{nonC} = \sum_{i=1}^{n} \frac{1}{R_{i,sh}} + \sum_{i=1}^{n} \frac{1}{R_{i,hd}} \tag{3}$$

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 AP. In this section, we will briefly introduce the CCMAC protocol, including how to choose the data rate and relay nodes, how cooperation is achieved between senders and helpers, and how the AP coordinates the senders to enable concurrent transmission. Finally, we discuss a few issues about the protocol.

4.1 Transmission Rate Detection and Helper Selection

Recall from equation 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 as the RBAR [5], the sender chooses the data rate based on the detected SNR. More precisely, when a sender joins the network, the transmission rate between the sender and AP, R_{sa}, is measured by the receiver, AP, and 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 any packets sent by the helper. Then, the sender gets an estimation of R_{hs}, and uses it as R_{hs}. The rate between each helper to the AP, R_{hd}, is measured by overhearing the transmissions from helper to AP and from which the data rate information is extracted.

Once the sender has all the values described above, it picks the candidate of helper nodes. The criteria of selecting the candidates is based on equation 4, which is an extension of equation 1. The difference between the two equations is that, in equation 4, the overhead is taken into consideration.

$$\frac{L}{R_{sh}} + \frac{L}{R_{hd}} + T_{HTS} + T_{overhead} < \frac{L}{R_{sd}} \tag{4}$$

In equation 4, L is the length of the data packet; the T_{HTS} is the time to send the HTS using the base rate, i.e. 1 Mbps for 802.11b; $T_{overhead} = 2 * (T_{SIFS} + T_{HD} + T_{PD})$, where T_{HD} and T_{PD} are the hardware circuit delay and propagation delay respectively; T_{SIFS} is the SIFS duration.

Once, the sender finds any of its neighbors satisfying equation 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. In contrast, if the transmission failed, the credit of the corresponding node will be decreased by 0.1. Once a node's credit 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 failure and node failure.

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.

4.2 The five different roles

Since CCMAC employs both cooperative communication and coordinated concurrent transmission, 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 nodes should act based on the current roles they are 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 toward the AP, which should be mostly protected. Any other transmission which may

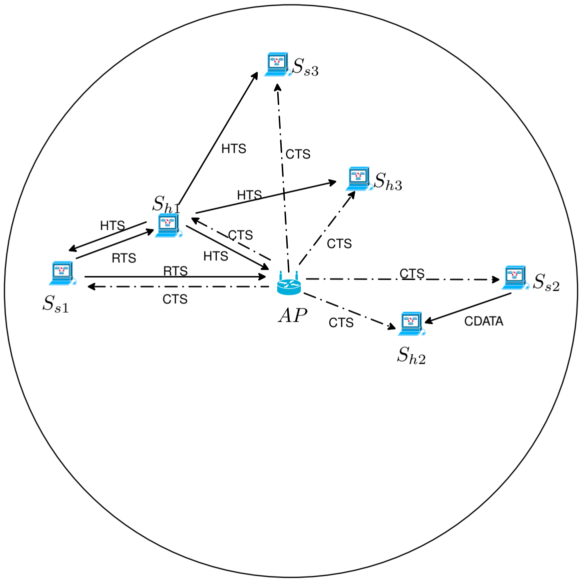

Figure 1: Network topology with seven nodes and the flow of messages.

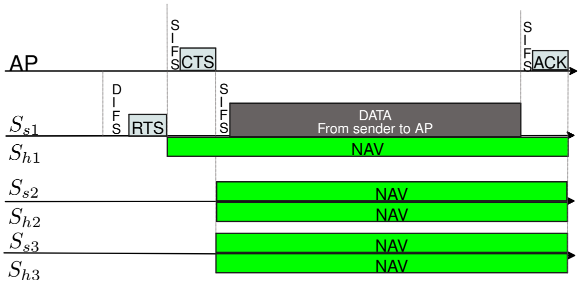

Figure 2: Example: message flow for basic mode.

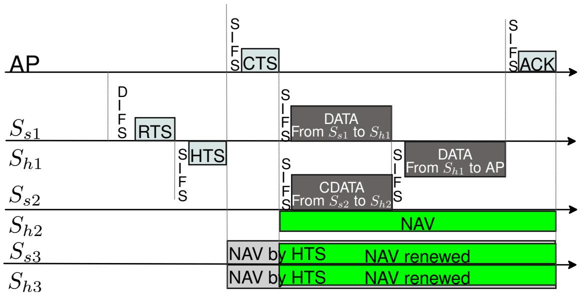

Figure 3: Example: message flow for enhanced mode

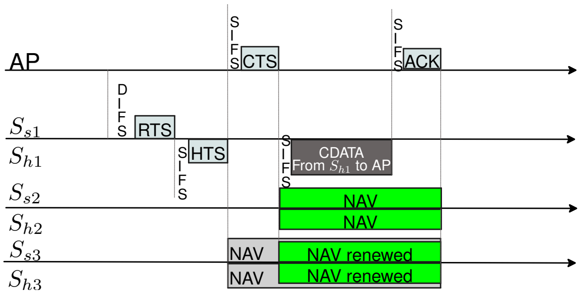

Figure 4: Example: message flow for half mode.

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. It is specified by the main sender, based on the main sender's helper table. Hence, there can be either zero 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 in charge of the coordination. It needs to know which nodes can send their packets without interfering the ongoing transmission (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 time utilizer (TU). As will be proved in theorem 1 later, there can be at most 4 TUs during one transmission, which means that the number of TUs can be zero to four. The TUs are the nodes selected by the AP (listed in AP's CTS) and tries 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 packet collision and 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 packet for the TU. 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 HTU's buffer. However, it will not affect the HTU's buffer which is reserved for other TUs.

4.3 The three transmission modes

In CCMAC, not all the roles will appear every time. It is decided by which mode of transmission is currently taking place. 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 7 wireless stations, one AP and 6 client stations. Station S_{s1} wins the contention and becomes the main sender (refer to the Figure 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, the normal WiFi transmission will be used. RTS/CTS messages are exchanged before the data packets, and the data packets is sent directly to the AP. Figure 2 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, two-hop transmission is used. 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. This tag number is to verify whether the packet is the correct packet later in the half mode. Figure 3 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 for the main sender is protected.

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; do nothing.
        else
            Apply enhanced mode; send data to helper.
        end
    else
        Apply base 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
```

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 the data is cached by helper. 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 4 shows the data flow sequence for the half mode.

To give a more detailed description of the algorithm, the pseudocode code is presented for the main sender (algorithm 1), helper (algorithm 2) and AP (algorithm 3) for the three different modes. We omit the pseudocodes for the TU and HTU, since their logic are straightforward.

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
```

4.4 Discussions

4.4.1 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 concerned about their own interest and do not care about the overall network performance. As a result, nodes may not want to cooperate if such cooperation only favors 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 to solve the problem and may induce extra cost to the network. In the case of CCMAC, where the 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, which means that, 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 the energy consumption. However, as shown in [11] that, sometimes the energy-per-bit experienced by the helper stations is decreased by participating in cooperation. This result is due to the reduction in idle energy consumption incurred by the helper, as it waits for its transmission opportunity while a slow node is occupying the channel. Hence, our conclusion is that, even if all nodes are selfish, they should be still willing to use the CCMAC protocol and cooperate accordingly.

4.4.2 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 base mode of CCMAC. Hence, for nodes which are not in AP's one-hop region, they can use the base 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.

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. 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 an incomplete information.

5.1 Finite state MDP and POMDP

A finite state MDP [13] is the basic model of a finite state POMDP. It defines a stochastic control process with components (S, A, P, R), which 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. If a system is formulated with this model, the decision made (i.e. the action selected) depends only on the current state, and we do not need to care about the history. A deterministic stationary policy π: $S \rightarrow A$ is a function that determines what action to take depending on the current state. The common objective for this model is to find the policy that 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.

A POMDP [15] [7] is a generalization of MDP, in which system states are not fully observable. This is a more realistic model for most real problems, since it is usually not possible to observe all the factors which may affect the system's behavior. However, this extension of MDP dramatically increases the complexity, which makes exact solutions virtually 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. A POMDP contains an underlying MDP, plus an observation space O and observation function Z. In an MDP, the agent has full knowledge of the system state, therefore, $S \equiv O$. In a POMDP, determining in which state the system is, 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: $S \times A \times O$, 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).

5.2 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. To model it 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 the information of which nodes are holding packets to send, plus the information about the wireless channel. To represent it mathematically, $S = \{K_1, K_2, K_3, K_4, B_1, B_2 \ldots B_N, Ch\}$. 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 1 in Section 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 vice versa. Lastly, Ch defines the channel characteristics.

Action ($a \in \{0, 1, \ldots, N\}$): 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 $K_1 \ldots K_4$, $B_1 \ldots B_N$. This is because, $K_1 \ldots K_4$ are the ID of the nodes, which have already been selected by the AP. For $B_1 \ldots B_N$, 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.

5.3 Using a RL algorithm to solve the AP coordination problem

Reinforcement Learning (RL) algorithms [16] 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 by interacting with the real world through a Monte Carlo-like method.

The standard approaches for solving MDPs are value iteration and policy iteration. Exact methods for solving POMDPs are highly 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 the doubly exponential rate.

An obvious approximation technique 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. For the AP coordination problem, we employ the IState-GPOMDP algorithm proposed by Aberdeen [1], which uses a policy gradient approach with the FSC to approximately solve this problem.

To use the FSC, we need to have an internal state I to represent the unobservable part of the real world. As proved in [3], 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:

$$\bigtriangledown\mu(a|\theta, o, i) = [\frac{\partial\theta_a^S}{\partial\theta_1^S}, \ldots, \frac{\partial\theta_a^S}{\partial\theta_{|A|}^S}] = [0, \ldots, 0, 1, 0, \ldots, 0] \quad (5)$$

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 6 below, are all zero except the i' element, which equals 1.

$$\bigtriangledown\omega(i'|\phi, o, i) = [\frac{\partial\phi_{i'}^S}{\partial\phi_1^S}, \ldots, \frac{\partial\phi_{i'}^S}{\partial\phi_{|I|}^S}] = [0, \ldots, 0, 1, 0, \ldots, 0] \quad (6)$$

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 $\beta$ and step size $\gamma$.
Set an arbitrary initial internal state $i_0$ and a random
starting point of $\phi$ and $\theta$, 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
        $z_{t+1}^\phi(S) = \beta z_t^\phi(S) + \frac{\bigtriangledown\omega(i_{t+1}|\phi, o_t, i_t)}{\omega(i_{t+1}|\phi, o_t, i_t)}$
        $z_{t+1}^\theta(S) = \beta z_t^\theta(S) + \frac{\bigtriangledown\mu(a_t|\theta, o_t, i_{t+1})}{\mu(a_t|\theta, o_t, i_{t+1})}$
        $g_{t+1}^\phi(S) = g_t^\phi(S) + \frac{1}{t+1}\left[r(t+1) z_{t+1}^\phi(S) - g_t^\phi(S)\right]$
        $g_{t+1}^\theta(S) = g_t^\theta(S) + \frac{1}{t+1}\left[r(t+1) z_{t+1}^\theta(S) - g_t^\theta(S)\right]$
        $t \leftarrow t + 1$
    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
```

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 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.

6. ANALYSIS

6.1 The maximum number of concurrent transmission

Before we calculate the number, 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.

Theorem 1. *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 5 that, node O represents the AP. We observe that, the transmission or interference area of any node (e.g. node P in Figure 5) in the unit circle equals to the intersection area of two circles with same radius r. In Figure 5, 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 5. The other is the non-relay 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 5. 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.

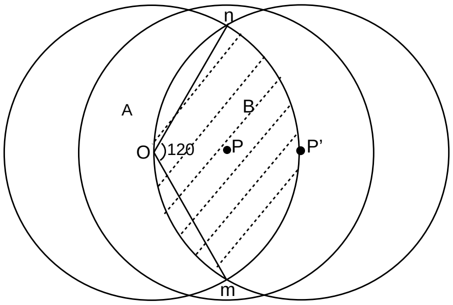

Figure 5: The intersection area and the relay area.

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, all points in the circle are covered by 2 sectors, which means that all the points are interrupted. Hence, 6 concurrent transmissions are not feasible. Finally, we find that 5 is a feasible value. One example of such a topology is to scatter the 5 senders at the corners of the largest regular pentagon, which fits the unit circle. □

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. For simplicity, we assume the loss rate for a control packet, like RTS/CTS/HTS/ACK, is zero. For nodes using the CCMAC protocol, the average transmission time for a node transmitting one packet is:

$$T_{cc}^{overall} = \frac{p_e}{p_e + p_h} T_{cc}^{en} + \frac{p_h}{p_e + p_h} T_{cc}^{half} \tag{7}$$

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.

$$T_{cc}^{en} = \frac{1-(1-p_f^P)^2}{(1-p_f^P)^2} T_{fail}^{en} + T_{succ}^{en} \tag{8}$$

$$T_{cc}^{half} = \frac{p_f^P}{1-p_f^P} T_{fail}^{half} + T_{succ}^{half} \tag{9}$$

$$\begin{aligned} T_{fail}^{en} = \; & T_{RTS} + T_{HTS} + T_{CTS} + \frac{L}{R_{s,h}} + \frac{L}{R_{h,d}} \\ & + T_{DIFS} + 4T_{SIFS} \end{aligned} \tag{10}$$

$$\begin{aligned} T_{fail}^{half} = \; & T_{RTS} + T_{HTS} + T_{CTS} + \frac{L}{R_{h,d}} \\ & + T_{DIFS} + 3T_{SIFS} \end{aligned} \tag{11}$$

$$T_{succ}^{en} = T_{fail}^{en} + T_{ACK} + T_{SIFS} \tag{12}$$

$$T_{succ}^{half} = T_{fail}^{half} + T_{ACK} + T_{SIFS} \tag{13}$$

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.

7. SIMULATIONS AND RESULTS

7.1 Simulation Setup

The simulation is done using OMNet++ [12] to simulate an IEEE 802.11b network. For this network, an AP is at the center of a unit circle with radius equals 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 [10], 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 [20] [17], 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$_4$. The WiFi is the legacy 802.11b DCF protocol with channel rate adaptation, i.e. the RBAR extension. CoopMAC is the protocol described in [10], and is the main comparison target for CCMAC. CCMAC is the protocol proposed in this paper. Note that, before each experiment, we let the AP learn how to coordinate for 10 mins before taking the results. The protocols of random$_1$ and random$_4$ have the same cooperation part as CCMAC. However, these protocols do not perform coordination. 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 to CCMAC.

7.2 Experiments

In the first experiment, there are 8 sender 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. The results are shown in Figure 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, CCMAC reaps the most benefit and it outperforms CoopMAC by around 15%; and it outperforms the random$_1$ and random$_4$ around 10%.

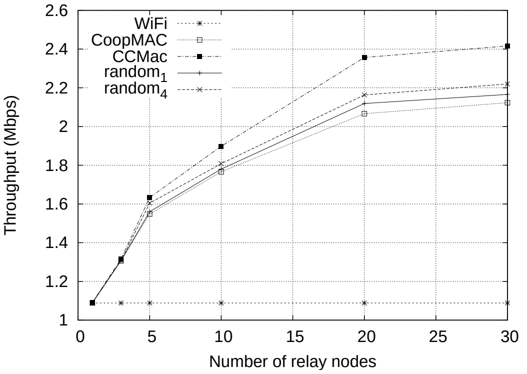

Figure 6: The average throughput achieved with different number of relay nodes.

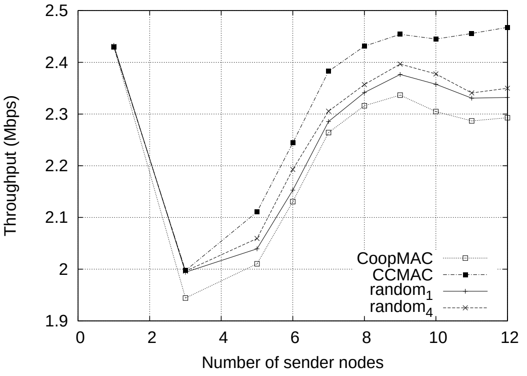

Figure 7: The average throughput achieved with different number of sender nodes.

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 to see the performance difference. The results are shown in Figure 7. From the graph, although the performance of all the protocols is affected by the senders' random location, the performance of CCMAC is still the best among all the four protocols. Clearly, 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 simultaneously. It shows that, when the number of nodes is increased, protocols with concurrent transmission capability, such as CCMAC, reap more benefits.

In the third experiment, we set the number of relay nodes to 15 and the number of senders as 8. We choose different packet sizes to see the performance of different protocols. The results are shown in Figure 8. Clearly, packet size is

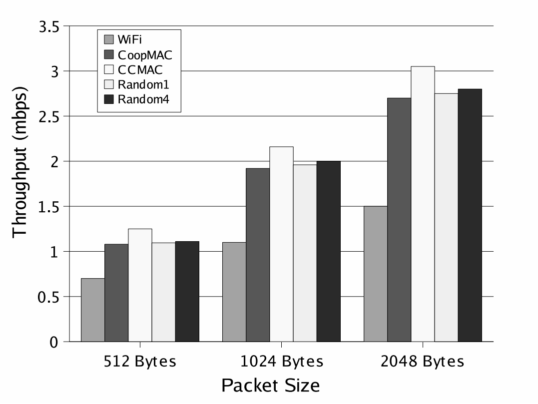

Figure 8: Average throughput achieved with different packet size.

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 smallest 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 cancelled 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 change of packet size has a larger effect on CCMAC than CoopMAC. The reason can be shown mathematically. Suppose the transmission time of sending a packet by CCMAC and CoopMAC is T_{cc} and T_{coop}, respectively. They have two components: C representing the constant overhead of CCMAC and CoopMAC, which are not related to packet length. Since the overheads of the two protocols are almost similar, we have $C_{cc} = C_{coop}$. The second component is the transmission time, which is related with the packet size. We define the coefficient k and use $k \times L$ to represent this. Clearly, $k_{cc} < k_{coop}$. 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.

8. CONCLUSION

In this paper, we explored the benefits of cooperation 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 has three different transmission modes: basic mode, enhanced mode and half mode. One of the modes is chosen based on the channel condition and the helper's status of whether a data packet have been cached. 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.

9. REFERENCES

[1] D. A. Aberdeen. Policy-Gradient Algorithms for Partially Observable Markov Decision Process, Ph.D Thesis, Australian National Unversity, March 2003.

[2] V. Bharghavan, A. J. Demers, S. Shenker, and L. Zhang. MACAW: A media access protocol for wireless LAN's. In *ACM SIGCOMM*, pages 212–225, 1994.

[3] B. Bonet. An ϵ-optimal grid-based algorithm for Partially Observable Markov Decision Processes. In *Proceedings of the International Conference on Machine Learning*, 2002.

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

[5] 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.

[6] IEEE. 802.11 standard http://standards.ieee.org/getieee802/802.11.html.

[7] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2), 1998.

[8] A. Kamerman and L. Monteban. WaveLAN-II: a high-performance wireless LAN for the unlicensed band. *Bell Labs Technical Journal*, 2(3), 1997.

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

[10] P. Liu, Z. Tao, S. Narayanan, T. Korakis, and S. S. Panwar. CoopMAC: A Cooperative MAC for Wireless LANs. *IEEE JSAC*, 25(2):340–354, 2007.

[11] S. Narayanan and S. S. Panwar. To forward or not to forward - that is the question. *Wirel. Pers. Commun.*, 43(1), 2007.

[12] OmNet++Community. http://www.omnetpp.org/.

[13] M. L. Puterman. Markov Decision Processes: Discrete Stochastic Dynamic Programming , John Wiley and Sons, NY, 1994.

[14] Status of IEEE 802.11n, IEEE 802.11 - Task Group n - meeting update http://grouper.ieee.org/groups/802/11/reports.

[15] E. J. Sondik. The Optimal Control of Partially Observable Markov Processes Over the Infinite Horizon: Discounted Costs. *Operations Research*, 101(1-2), 1978.

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

[17] Q. Zhang and S. Kassam. Finite-state Markov model for Rayleigh fading channels. *IEEE Trans. Commun*, 47(11):1688–1692, 1999.

[18] H. Zhu and G. Cao. On improving the performance of IEEE 802.11 with relay-enabled PCF. *Mobile Networks and Applications*, 9(4), 2004.

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

[20] 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*.