Performance evaluation of channel selection algorithm for multi-channel MAC protocol in ad hoc networks

by

© Bin Han

A thesis submitted in partial fulfillment of

the requirements for the degree of Doctor of Engineering

Department of Computer Science

Graduate School of Engineering

Gunma University

November 2012

Abstract

This thesis aims to provide an approach that is to investigate channel selection algorithm for increasing the performance of ad hoc networks. Although our channel selection algorithms are very simple, multi-channel MAC protocol that employs our channel selection algorithms are effective for increasing the performance of ad hoc networks.

Acknowledgements

This work would not have been possible without the help from many people. I would like to thank them at here.

First of all, I must express my deepest gratitude to my professor Ken'ichi Kawanishi. His help, feedback, and support were absolutely crucial for the successful completion of this thesis. I am considering myself extremely fortunate to being his student.

I gratefully thank Professor Yoshikuni Onozato, Professor Yoichi Seki, Professor Kenetsu Fujita and Professor Ushio Yamamoto for being my thesis reviewers.

I am also deeply grateful to Li Wang who was a doctor course student of Kawanishi Lab. He gives a big help on Network Simulator 2 (NS2). Because of his help, I save a lot of time on studying NS2. I am thankful to all past and current students of Kawanishi Lab. and Fujita Lab. for the great working atmosphere.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

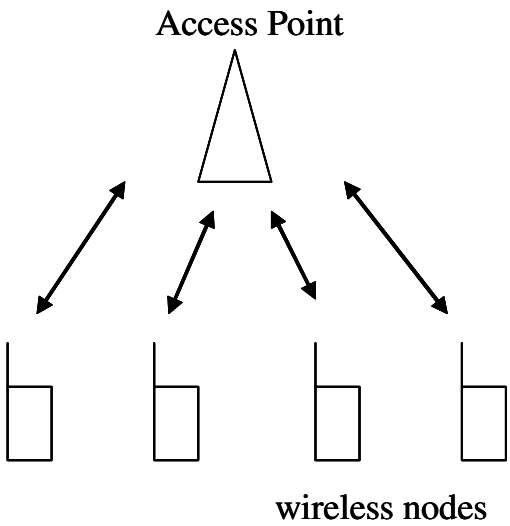
For the last twenty years, Wireless Local Area Networks (WLANs) have experienced explosive growth. WLANs were designed as data transmission systems to ensure the connection which is independent from the physical location of computer peripherals making up the network and the connection uses wireless connection instead of a wired infrastructure.

WLANs standards may support two models of operation: an infrastructure based mode and an ad hoc mode. In the infrastructure based mode, the communications between two nodes are done through the Access Point (AP). On the contrary, the ad hoc mode is a peer-to-peer mode where each node may act as a router for the others. WLANs have a single hop wireless network with a small area of coverage. However, the latter mode is used to allow multi-hop communications between nodes. Among those standards we could find the de-facto WLANs standard IEEE 802.11 [1]. For the brief introduction of this WLANs standard sees Appendix A.

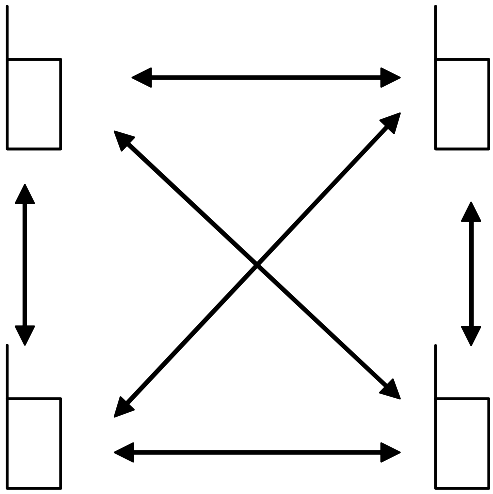
Ad hoc networks are complex distributed systems that consist of wireless nodes that can freely and dynamically self-organize. In this way they form arbitrary and temporary "ad hoc" networks topologies, allowing wireless nodes to seamlessly interconnect in areas with no pre-existing infrastructure. Therefore, ad hoc networks are the practical and interesting networks connection solution that provides flexibility, low deployment and usage cost for wireless connections.

Meanwhile, in today's networks, users have the stronger demand for the high performance networks than before, since users need these networks to run their applications, such as videos, games and so on. Therefore, how to increase the performance of networks has became a hot topic and this trend also happens in ad hoc networks.

One of simple ways to increase the performance of ad hoc networks is to use multi-channel Medium Access Control (MAC) protocol [2]. This is because, by non-overlapping



(a) Four wireless nodes connected with each other through AP in an infrastructure based mode.



(b) Four wireless nodes connected with each other in an ad hoc mode.

Figure 1.1: Infrastructure based mode and ad hoc mode

multiple channels, a pair of source and destination nodes that uses this protocol is assigned the exclusive use of one channel to exchange packets. That makes concurrent and successful data transmission without interference with each other. Therefore, multi-channel MAC protocols have a potential way to increase the performance of ad hoc networks significantly.

In recent research of multi-channel MAC protocol for ad hoc networks, it is assumed that

the number of packet interfaces on each node is less than the number of channels. Thus, designing a multi-channel MAC protocol for ad hoc networks has two main challenges; one is to design a medium access mechanism for assigning a channel on each wireless node, and the other is to design a suitable channel selection algorithm for data transmission. A number of multi-channel MAC protocols to design medium access mechanism have been proposed and much investigated in the past, such as [3]–[21]. However, to design a suitable channel selection is still an important open issue for the multi-channel MAC protocol.

On the other hand, the Transmission Control Protocol (TCP) is typically used to provide transport-layer service on the top of the MAC protocol. This approach allows wireless nodes of ad hoc networks to integrate seamlessly with existing networks. A number of works [22]–[27] have shown that TCP can perform poorly when the underlying networks use the MAC protocol. Therefore, designing an adapted TCP mechanism for multi-channel MAC protocol to increase the performance of ad hoc networks is also investigated in [28]. However, to design an adapted TCP mechanism that employs channel selection algorithm is another interesting point of view to improve the performance of ad hoc networks.

1.1 Thesis contributions and organization

In this thesis, we focus on the impact of multiple channels on the performance of ad hoc networks, especially our viewpoints from the channel selection algorithm on MAC protocol and TCP mechanism on Transport protocol. The chapters of the thesis can be classified into the following of three parts.

- The first part deals with the channel selection algorithm over multi-channel MAC protocol that consists of Chapter 2.

  In this part, we consider two issues for the channel selection algorithm. One issue is the way of selecting which channel is used for data transmission between pairs of nodes. The other issue concerns how long the channel should be used between the same pair of nodes. Because wireless links in ad hoc networks might be disconnected due to, for example, node mobility, these issues significantly influence the efficiency of multiple channels and hence the performance of multi-channel MAC protocols. We believe that solving two issues can lead multi-channel MAC protocol to increase the performance of ad hoc networks effectively.

  The main contribution in this part is to propose a simple and efficient channel selection algorithm for the multi-channel MAC protocol based on Extended Receiver Directed Transmission (xRDT) protocol [3] which only uses one packet interface. Briefly, our channel selection algorithm uses the active channel for data transmission until the amount of data packets reaches a threshold, at which point it selects one of the

available channels *except the active channel*. Therefore, we call our channel selection algorithm, Conditionally Randomized Channel Selection, and we abbreviate the algorithm as CRCS. Note that the main difference between xRDT and CRCS is their channel selection algorithms. Although CRCS is simpler than the channel selection algorithm adopted in xRDT, CRCS is a more effective channel selection algorithm than that of xRDT to increase the performance of ad hoc networks as well as to keep the load balance of all channels. Details of the proposed channel selection algorithm and xRDT protocol (including its channel selection algorithm) are explained later in this chapter.

Also, in this part, we use computer simulation with the simple topology in single collision domain and multiple collision domain to evaluate the performance of CRCS. In a single collision domain, all nodes are within the transmission range of one node. On the other hand, in a multiple collision domain, some nodes may be within the transmission range of a node and others may be out of the transmission range of the node. In order to show that CRCS is an effective channel selection algorithm to increase the performance of ad hoc networks and to keep the load balance of all channels, we compare the performance of CRCS with those of the other channel selection algorithms based on xRDT, such as Randomized Channel Selection (RCS) [1] based on xRDT, and some multi-channel MAC protocols.

- The second part deals with the investigation of channel selection algorithm by using game theory that consists of Chapter 3.

  In Chapter 2, we designed two simple stochastic channel selection algorithms called CRCS and RCS for multi-channel MAC protocol. It is shown that CRCS can provide higher throughput than RCS, especially in multiple collision domain. This is somewhat surprising because the difference between CRCS and RCS is a little; Rather than selecting a new channel from all available channels as in RCS, CRCS selects a new channel from available channels *except the active channel*. Because the analysis is based on computer simulation, it is not clear that why the wireless node should set equal probability when selecting a new channel, and what the reason is that makes the difference of throughput between CRCS and RCS.

  The purpose of Chapter 3 has twofold. First, we investigate rationale of the equal probability distributions in CRCS and RCS by using game theory [29]. To this end, we assume wireless nodes as selfish agents who are trying to improve their throughput. We also assume that wireless nodes are *rational* and *intelligent*. The objective of a rational

[1] RCS has the same mechanism with CRCS except to select a new channel from all available channels with equal probability. We will also explain RCS in Chapter 2.

wireless node is to maximize the expected value of his own payoff, which is measured in some utility scale. An intelligent wireless node can make any inferences about the situation, if he knows everything that we know about the game. By formulating stochastic channel selection algorithms as games, we show that the equal probability distributions in CRCS and RCS are the stable operating points of the probability distribution for selecting channels at which all selfish wireless nodes agree to operate. Second, we show that CRCS can provide the higher expected payoff than RCS, which is the reason of the surprising result that CRCS can give the higher performance than RCS. We provide some numerical results by computer simulation to support our assertion.

- The third part deals with TCP over multi-channel MAC protocol that consists of Chapter 4.

  In this part, we tune TCP's congestion window dynamics to enhance the TCP performance in multi-channel MAC protocol that employs CRCS as its channel selection algorithm. Many researches in the past revealed that TCP can poorly perform with IEEE 802.11 MAC protocol, especially in a scenario of multi-hop wireless environment, due to the TCP instability [30] and the unfairness [31] problems. The TCP instability problem is that the throughput of a TCP flow fluctuates severely and even frequently drops to zero. The unfairness problem means that some flows tend to dominate the channel and the other flows are starved when there are several TCP flows competing in the network. We believe that the TCP instability and unfairness problems still happen in multi-channel ad hoc networks. To mitigate thees problems, we pay attention to the fact that more than one packet cannot be *in-flight* on a single wireless link at once as is pointed out by [32]. Moreover, we take into account of the theoretical and simulation-based analyses in [32] suggesting that the 'optimal' value of the upper bound of congestion window size is relatively small for a single TCP flow in a line topology. Relying on the observations and analyses, we modify the TCP's congestion window dynamics. We increase the congestion window gradually so as not to overshoot the upper bound frequently.

  We also introduce a cross-layer scheme between TCP and MAC layers. It is known that TCP's congestion control and retransmission mechanism do not collaborate well with wireless MAC layer protocol, which causes the performance degradation in multi-hop ad hoc networks. In particular, the retransmission timeout in TCP layer greatly degrades the performance and is not preferable. In this part we propose a scheme to reduce the retransmission timeout events in TCP layer. Our scheme allows MAC protocol to send an immediate-retransmission message to TCP layer when the data packet drops at MAC layer. We evaluate the TCP throughput and fairness perfor-

mance for ad hoc networks by computer simulation. The results of simulation show that our multi-channel MAC protocol that employs CRCS as its channel selection algorithm can improve the TCP performance significantly.

Chapter 2

Channel Selection Algorithm for Multi-Channel MAC Protocol

The Medium Access Control (MAC) protocol that uses non-overlapping multiple channels, called the multi-channel MAC protocol, was proposed in order to increase the performance of ad hoc networks. Since the number of packet interfaces on each node is less than the number of channels in ad hoc networks in general, the node needs to select a suitable channel for data transmission. This means that the multi-channel MAC protocol must be provided with a good channel selection algorithm. In this chapter, we design a simple channel selection algorithm called Conditionally Randomized Channel Selection (CRCS) based on Extended Receiver Directed Transmission (xRDT) protocol that only uses one packet interface. Briefly, CRCS uses the active channel for data transmission until the amount of data packets reaches a threshold, at which point it selects one of the available channels other than the active channel. Although CRCS is a very simple channel selection algorithm, by using network simulator we find that CRCS is effective to increase the performance of ad hoc networks and to keep the load balance of all channels compared to the other channel selection algorithms.

Note: The materials of this chapter have been appeared in [33].

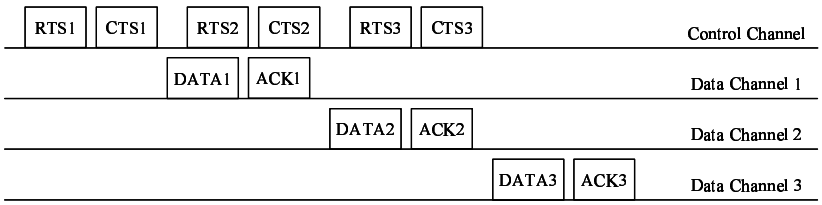


Figure 2.1: Dynamic Channel Allocation (DCA).

2.1 Background

Multi-channel MAC protocols are designed to implement on the number of network interfaces available in one node as follows: multiple interfaces, a node that can access multiple channels simultaneously, and single interface, a node that can only access one channel at a time. Note that the number of interfaces in each node is less than the number of channels in ad hoc networks and the interface is capable of switching from one channel to another. There are two categories in multi-channel MAC protocols, i.e., *asynchronous* and *synchronous* schemes.

2.1.1 Asynchronous scheme

An asynchronous approach of multi-channel MAC protocols using multiple interfaces is Dynamic Channel Allocation (DCA) [4] protocol. In DCA, each node has two interfaces. One interface is assigned a channel dedicated to control messages and the other one is used for data packets. It implies that DCA can listen on both the control and data channels simultaneously. We illustrate DCA protocol in Fig. 2.1. As shown in this figure, the control messages of Request-to-Send (RTS) and Clear-to-Send (CTS) are exchanged on the control channel, on the opposite data packet (DATA) and the control message of acknowledgement (ACK) are transmitted on the data channel. The similar multi-channel MAC protocols as the same approach like DCA using multiple interfaces are investigated in [5]–[11]. On the other hand, the asynchronous approaches of multi-channel MAC protocols using single interface have also been investigated in [12]–[14]. In this approach, the interface is capable of switching the channel between the control channel and the data channel.

The drawback of DCA is the hidden terminal problem [1] of multi-channel ad hoc networks. This is because when a node is active on a data channel, it is unable to learn the situation of its neighboring channel. In other words, the node loses the channel information

[1] In Appendix A, we explain the hidden terminal problem of ad hoc networks under the single-channel environment.

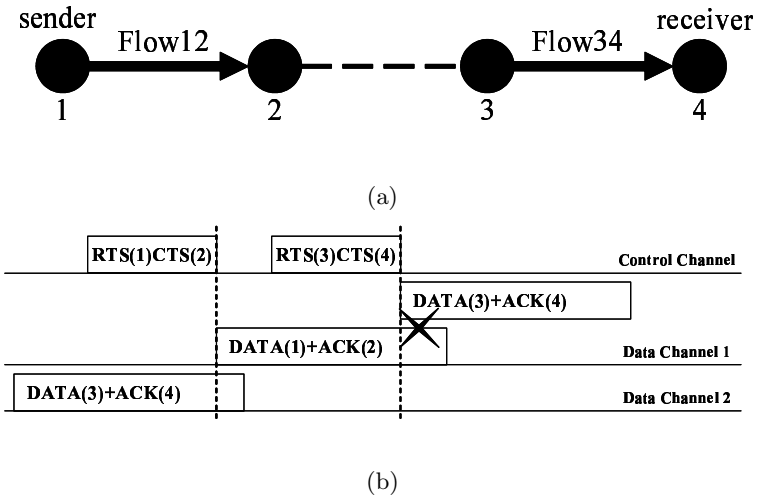


Figure 2.2: Illustration of the hidden terminal problem of DCA.

of its neighboring nodes. Therefore, the node may inadvertently choose the same channel when it begins to exchange its next data packet. For example, in Fig. 2.2, suppose that $Flow12$ transmits data packets on data channel 1, while $Flow34$ exchanges RTS/CTS messages on the control channel as shown. Since $Flow34$ has not heard the reservation of $Flow12$, it may select data channel 1. In this case, the collision happens on the data channel 1. Furthermore, DCA has another problem. When the number of channels and transmission flows are large, the control channel is filled with all the negotiation of control messages and too much contention will cause bottleneck and saturation on the control channel. The solutions for this problem to increase channel utilization are proposed in [10, 11].

The major advantage of the asynchronous approach of multi-channel MAC protocols using multiple interfaces is that it does not require time synchronization on the same channel. On the other hand, the disadvantage of this approach is that it increases the cost for multiple interfaces.

2.1.2 Synchronous schemes

A synchronous approach of multi-channel MAC protocols using single interface is Multi-channel MAC (MMAC) [15] protocol. This protocol does not need a separate control channel. Instead, it utilizes an Ad hoc Traffic Indication Message (ATIM) window in the common channel to negotiate channels using one interface. The ATIM window is the time synchronization phase when 802.11 Power Saving Mechanism (PSM) is applied. We illustrate MMAC protocol in Fig. 2.3. In this example, we assume that the common channel of MMAC is channel 1. Some protocols that can be viewed as extension of MMAC are investigated in [16]–[18] to increase the energy efficiency of MMAC. The basic idea of their

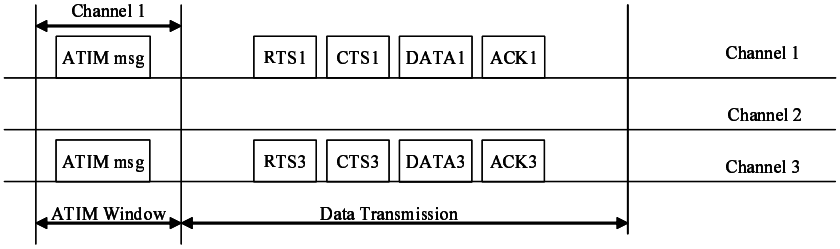


Figure 2.3: Multi-channel MAC (MMAC).

protocols is to arrange the idle node into power saving mode to avoid unnecessary power consumption.

MMAC can keep the load balance of all channels by using ATIM window to assign all possible channels. In other words, MMAC can alleviate the problem that the channel becomes to be the bottleneck. However, the overhead on the common channel affects the performance of ad hoc networks when the duration of ATIM window has to be long enough to accommodate all nodes in the neighborhood.

Another synchronous approaches of multi-channel MAC protocols also using single interface are the channel hopping protocols. In Channel Hoping Multiple Access (CHMA) [19] and Channel Hopping multiple Access with packet Trains (CHAT) [20], all nodes have a common hopping sequence. The transmitter and the receiver stop hopping when they are doing data transmission, and rejoin the common hopping sequence afterward. In Slotted Seeded Channel Hopping (SSCH) [21], it does not use the common hopping sequence. Instead, the number of hopping sequences is equal to the number of channels. In SSCH, each node randomly selects a sequence. Only the transmitter and the receiver can do data transmission when their hops are onto the same channel

The advantage of the synchronous approach of multi-channel MAC protocols using single interface is that it requires only one interface per node for reducing the hardware cost. However, this approach requires time synchronization among all of nodes. It is well known that synchronous scheme is not suitable for mobile networks.

2.1.3 Extended Receiver Directed Transmission (xRDT) protocol

Compared to all of those multi-channel MAC protocols, xRDT [3] is a multi-channel MAC protocol based on busy tones. In xRDT, two interfaces are assumed; one is a packet interface used for data transmission and the other is a tone interface for busy tone. xRDT assumes that all of channels have the same bandwidth. Each channel bandwidth is separated into one wide bandwidth as the data channel for transmitting data packets and one narrow

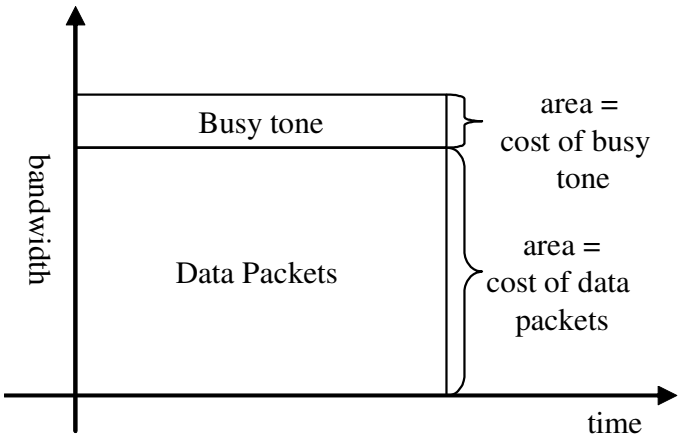


Figure 2.4: Each channel's bandwidth is separated into one wide and narrow bandwidth.

bandwidth as the signal channel for busy tone as shown in Fig. 2.4. A different busy tone is associated for each channel. Therefore, xRDT using two interfaces accesses one channel at a time. It is considered that tone interfaces are simple to implement so that each node in xRDT must have only an additional tone interface, rather than two packet interfaces as in DCA.

In xRDT, every node listens to a channel when the node has no data packet to transmit. The channel on which the node is tuned when there is no data packet is called *quiescent channel*. If a node wants to send data packets to a neighboring node, then it will switch its interface to the quiescent channel of the intended *receiving* node. Then, the node senses the busy tone associated with the quiescent channel. If the busy tone is not detected, then the transmitter keeps on scanning channels using RTS messages until the transmitter can get the busy tone signal from the receiver. When the receiver can receive RTS, it turns on the busy tone associated with the quiescent channel, instead of sending CTS. By sensing on the busy tone, any other potential transmitters defer to send data packets. When the transmitter hears the busy tone on the quiescent channel, it starts to transmit data packets to the receiver. On the other hand, potential transmitters go into the back-off duration like IEEE 802.11. On completing the data packet transfer, the receiver turns off the busy tone. After an appropriate inter-frame spacing, the receiver again turns on the busy tone as an acknowledgment.

Figure 2.5 shows a workflow of xRDT. When *node0* hears the busy tone on the quiescent channel, it starts to transmit data packets to *node1*. At the same time, *node2* switches its interface to *ch2* for receiving data packets. Then, *node1* and *node2* can make concurrent data transmissions without interference with each other. On the other hand, since *node3* cannot know that *node2* changes to *ch2* and cannot hear the busy tone on its quiescent

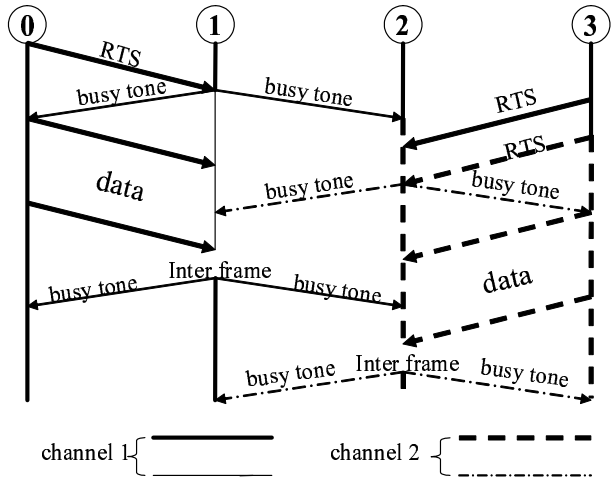


Figure 2.5: The workflow of xRDT, where the distance between neighboring nodes is 200m, the transmission range is 250m and the carrier sensing range is 550m.

channel, it scans the next channel using RTS message until *node3* gets the busy tone from *node2*.

xRDT exploits busy tone because it can alleviate the hidden terminal problem in multi-channel environment. The hidden terminal problem in multi-channel ad hoc networks happens because a node cannot hear data packet transmissions taking place on the different channel when it is listening on a particular channel. By using busy tone associated with the quiescent channel, neighboring nodes are able to monitor the channel situation continuously. Moreover, by choosing the quiescent channel on a *receiver*, a transmitter knows whether the channel is used on the receiver. Therefore, the hidden terminal problem in multi-channel environment can be taken care of and the contention of data packets on wireless channel can be reduced in xRDT. By also using busy tone, we expected that the channels can be efficiently utilized since the busy tone can be viewed as if it is an implicit receiver-side reply whose size is small enough. In Appendix B, we use Markov model to show that using busy tone can alleviate hidden terminal problem and increase the performance of multi-channel ad hoc networks.

As summary, we show existing multi-channel MAC protocols in Table 2.1.

2.1.4 Channel selection algorithms

One classic channel selection algorithm for multi-channel MAC protocols is randomized channel selection algorithm. In DCA, the transmitter randomly selects one channel for

Table 2.1: Comparison of existing multi-channel MAC protocols

Protocols	Hardware requirement	Synchronous required	Channel bottleneck	Hidden terminal
Multi-channel CSMA [5] and [6]	More than two interfaces	No	Not alleviate	Not alleviate
DCA [4]	Two interfaces	No	Not alleviate	Not alleviate
DCA-PC [7]	Two interfaces	No	Not alleviate	Not alleviate
DPC [8]	Two interfaces	No	Not alleviate	Not alleviate
COMMAC [9]	Two interfaces	No	Not alleviate	Not alleviate
Pipelining DCA [10]	Two interfaces	No	Alleviate	Not alleviate
SAM-MAC [11]	Two interfaces	No	Alleviate	Alleviate
xRDT [3]	Two interfaces	No	Alleviate	Alleviate
Bi-MCMAC [12]	One interface	No	Not alleviate	Alleviate
AMCP [13]	One interface	No	Not alleviate	Alleviate
CAM-MAC [14]	One interface	No	Alleviate	Not alleviate
MMAC [15]	One interface	Yes	Alleviate	Alleviate
PSM-MMAC [16]	One interface	Yes	Alleviate	Alleviate
TMMAC [17]	One interface	Yes	Alleviate	Alleviate
CMMP [18]	One interface	Yes	Alleviate	Alleviate
CHMA [19]	One interface	Yes	Not alleviate	Alleviate
CHAT [20]	One interface	Yes	Not alleviate	Alleviate
SSCH [21]	One interface	Yes	Alleviate	Alleviate

transmitting data packets. This channel information will be encapsulated into RTS that is sent to the receiver on the control channel. When the receiver gets RTS, it sends CTS back to the transmitter to confirm the channel is reserved.

On the other hand, MMAC uses the load based channel selection algorithm. Each node maintains a data structure called the Preferable Channel List (PCL), that indicates which channel is preferable to use for the node. PCL records the usage of channels inside the transmission range of the node. Based on this information, the channels are categorized into three states: high preference, medium preference, and low preference. The node prefers to select higher preference rather than lower preference.

In xRDT protocol, the channel is changed depending on the traffic load. The algorithm of xRDT protocol selects the channel that has the lowest load from the set of available channels. The lowest load channel in xRDT means the channel of which the idle time, i.e., duration not used for data transmission, is the longest one.

In [6], the authors consider the interference power on all channels to select the best channel. At the time of transmission, the transmitter measures the signal powers (the level of the carrier signal) on all channels and selects the channel that has the minimum carrier power. The same authors suggest that each node can select the clearest channel [34]. The clearest channel has the maximum Signal to Interference plus Noise Ratio (SINR) at the receiver. This channel selection algorithm is expected to provide the highest probability of successful data transmission [2]. However, these channel selection algorithms are too complex for ad hoc networks.

The main task of those channel selection algorithms is only to increase the throughput performance of ad hoc networks. However, they do not take care of keeping the load balance of all channels. The gap in existing solutions motivated us to propose a simple and efficient channel selection algorithm for multi-channel MAC protocol that not only increases the throughput performance of ad hoc networks but also keeps the load balance of all channels. The approach of the channel selection algorithm is the main focus of this chapter.

2.2 Conditionally Randomized Channel Selection (CRCS)

It is intuitively obvious that increasing the concurrency of data transmissions in multi-channel MAC protocol including xRDT heavily depends on how efficiently the quiescent channel is assigned to each of nodes. In our proposed multi-channel MAC protocol, we also assume that each node can change its quiescent channel when it receives busy tone on its quiescent channel as xRDT does. Compared to xRDT, we propose a dynamical algorithm

[2] In this chapter, a data transmission of multi-channel MAC means a cycle that includes a handshake procedure during control messages and the data packet, such as RTS/CTS/DATA/ACK in IEEE 802.11.

of selecting quiescent channels. The basic idea of our algorithm is to change the quiescent channel randomly among available channels other than the active quiescent channel after the amount of data packets reaches a threshold.

Suppose that there are $M > 1$ available channels. For each channel, we denote by th_c the threshold for channel c. We assume that th_c is pre-determined and a constant which may be different between channel-to-channel. We also introduce a time-dependent variable $r_c(t)$ for channel c as the cumulative number of data packets that the receiver received on the current quiescent channel. It is assumed that the receiver can monitor and count the number of data packets received.

At the time when the quiescent channel should be changed, the receiver sets $r_c(t) = 0$, and then starts to measure $r_c(t)$ until a new quiescent channel is selected. The acitve quiescent channel c is changed for the first time when $r_c(t)$ is equal to or exceeds the threshold th_c. For example, the quiescent channel is updated every time when the receiver received one data packet if the threshold is equal to one.

A new quiescent channel is selected from available channels other than the acitve quiescent channel at random. For example, if the acitve channel is channel 1, then the remaining $M - 1$ channels (channels 2 through M) are equally selected as a new quiescent channel with probability $1/(M - 1)$. The receiver then informs the new quiescent channel to the neighboring nodes by using broadcast through the old quiescent channel. After informing the new quiescent channel, the receiver begins to listen on the new quiescent channel.

CRCS is summarized as follow.

```
initialize the receiver's quiescent channel (e.g., channel c)
loop
    r_c(t) = 0
    while r_c(t) < th_c do
        update r_c(t) when a new data packet is received
    end while
    choose one element c' in {1, 2, ..., M} \ {c} at random
    broadcast the new quiescent channel c'
    switch the interface to the new quiescent channel c'
    c = c'
end loop
```

Figure 2.6 shows the process of CRCS. The transmitter sends RTS to the receiver. If it does not get reply from the receiver, it changes another channel to send RTS. When the receiver receives RTS, it turns on its tone interface to broadcast busy tone. When the transmitter hears the busy tone, it starts to send data packets. After the receiver receives data packets, it starts to count the number of data packets received, $r_c(t)$. If $r_c(t)$ is less than th_c, the receiver still uses this channel to make data transmission. If $r_c(t)$ is equal to

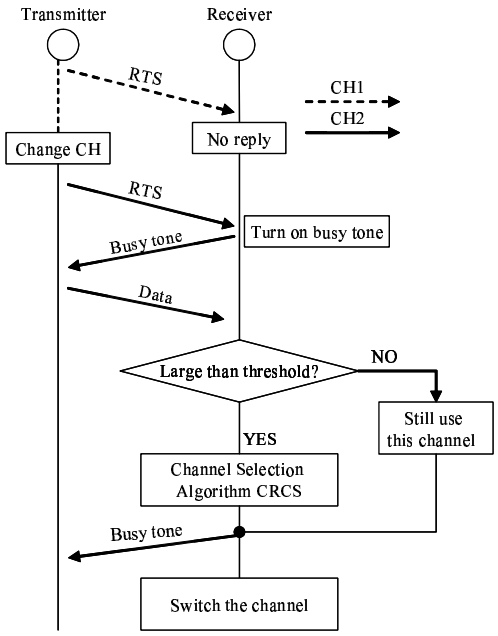


Figure 2.6: The process of our channel selection algorithm.

or larger than th_c, the receiver decides to change a new channel as a quiescent channel to make data transmission. Then, the receiver turns on the busy tone as acknowledgment. At the same time, the receiver switches its packet interface on the new channel.

CRCS has some advantages. First, it increases and balances utilization across all available channels, not fixed on one channel. Secondly, the overhead of switching the quiescent channel can be reduced in our algorithm by carefully choosing the threshold. Thirdly, it can reduce that the quiescent channel becomes to a bottleneck. Moreover CRCS could solve the issues for which channel and how long the channel should be used for data transmission between the pair of nodes by selecting the channel *conditionally randomized* and setting the threshold at the receiver.

2.3 Performance evaluation

In this section, we evaluate the performance of our channel selection algorithm by network simulator ns-2 [35]. We compare the proposed channel selection algorithm, CRCS, with Constant Channel Selection (CCS) and Randomized Channel Selection (RCS) algorithms based on xRDT. CCS is used as the most baseline channel selection algorithm. In CCS, once the receiver selects one channel, it will never change the channel again. RCS is also as the baseline channel selection algorithm. In RCS, the receiver simply selects a channel

uniformly at random from the set of all available channels. Note that the receiver in CRCS also selects a channel uniformly among the all available channels but the acitve channel. We also compare the performance of CRCS with xRDT to Dynamic Channel Allocation (DCA) [4], Multi-channel MAC (MMAC) [15], and the original xRDT [3].

As mentioned earlier, the main goal of our channel selection algorithms is to increase the performance of ad hoc networks and to keep the load balance of all channels. We employ three metrics to evaluate the performance of our proposed channel selection algorithm under multi-channel MAC protocol.

- *Average aggregate throughput* over all flows in ad hoc networks. CRCS under multi-channel MAC protocol is expected to increase the performance of ad hoc networks by exploiting multiple channels. This metric will directly show that whether CRCS under multi-channel MAC protocol achieves our goal.

- *Load balance index* over all channels as the metric in ad hoc networks. The load balance index η_i for channel i is defined by $\eta_i = X_i / \sum_{j=1}^{M} X_i \, (i = 1, 2, \ldots, M)$, where X_i is the number of received data packets on the channel i and we denote by M the number of available channels. The load balance index takes the value between 0 and 1. When η_i is equal to $1/M$ for all $i = 1, 2, \ldots, M$, the load balance index is the best.

- *The Jain's fairness index* over all flows in ad hoc networks. The Jain's fairness index ξ [36] is defined by $\xi = (\sum_{i=1}^{N} T_i)^2 / N \sum_{i=1}^{N} T_i^2$, where T_i is the throughput of connection i, and N is the total number of connections. The fairness index takes the value between 1 and $1/N$. When ξ is equal to 1, the fairness is the best. Oppositely, the fairness is the worst if ξ is equal to $1/N$.

2.3.1 Simulation model

We assume the following common parameters in each simulation; all the radio parameters are ns-2 defaults; the bit rate for each channel is $2Mbps$; the radio propagation mode is the two-ray ground model with transmission range $250m$; carrier sensing and interference ranges are both $550m$; the queueing buffer size is 50 packets on all of nodes; the number of nodes is 16 and 36 nodes in single collision domain and multiple collision domain; we randomly select half of the nodes as sources and the others as destinations; each source nodes generates and transmits constant-bit rate (CBR) traffic to each destination node. For each simulation, we examine the average aggregate throughput, the load balance index, and the Jain's fairness index by varying different channel selection algorithms based on xRDT and different multi-channel MAC protocols. Each simulation was performed for duration of 500 seconds. Each data point in the result graphs is an average of 50 runs.

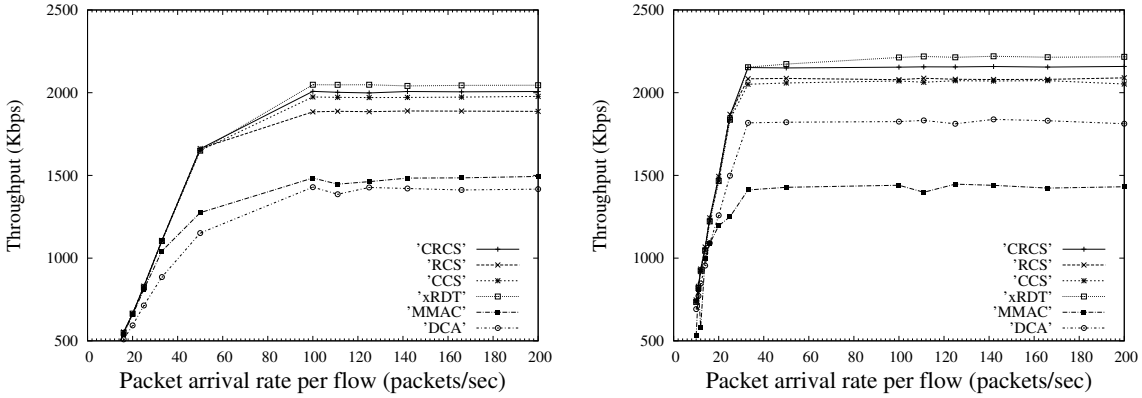
In the graphs, the curves labeled as "DCA", "MMAC", and "xRDT" indicate multi-channel MAC protocols of DCA, MMAC, and xRDT protocols. The curves labeled as "CRCS", "RCS", and "CCS" indicate channel selection algorithms by using Conditionally Randomized channel selection, Randomized Channel Selection, and Constant Channel Selection based on xRDT protocol.

2.3.2 Single collision domain

We evaluate the average aggregate throughput of different multi-channel MAC protocols in 16-node and 36-node topologies as increasing input packet arrival rate, when the total of three and six channels are used. As shown in Fig. 2.7, all of multi-channel MAC protocols yield similar performance until the arrival rate is less than 20 packets/sec because the load is too low to exploit the multiple data channels. If we increase the arrival rate more than 20 packets/sec, then difference of the way of handling multiple data channels between multi-channel MAC protocols significantly influences the average aggregate throughput. In fact, we can observe that multi-channel MAC protocols based on xRDT attains a higher average aggregate throughput than MMAC and DCA. Since DCA separates one channel to only exchange the control messages, DCA can only use two channels to transmit its data packets. This clearly reduces the utilization of multiple data channels and therefore DCA deteriorates the performance. Although MMAC can use all of available channels, too much overhead of negotiating channel happens periodically on the common channel that also reduces the utilization of multiple data channels. Comparing to the two multi-channel MAC protocols, xRDT and the xRDT-based multi-channel MAC protocols efficiently utilize the data channels, leading to increase the average aggregate throughput, because all of data transmissions are spread over on all of available channels.

We note that xRDT and the xRDT-based protocols can achieve different average aggregate throughputs because they have different channel selection algorithms. Although xRDT provides the highest throughput, we should notice that the channel selection algorithm in xRDT is complicated to find the lowest load channel for ad hoc node. Comparing to xRDT, CRCS is a simple way for ad hoc networks and provides a good throughput close to xRDT. More interestingly, we find that CRCS can provide a higher throughput than RCS. This is because RCS has a possibility that makes some pairs of transmissions select the same channel consecutively or prefer to select one channel. On the other hand, CRCS always selects a channel other than the active channel. Under this algorithm, CRCS is faster than xRDT to make all of channels distributed equally on all pairs of transmissions.

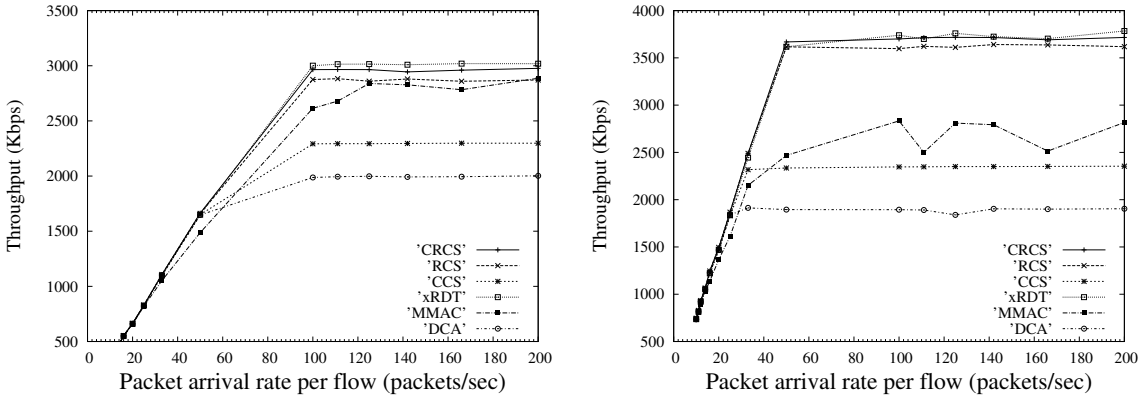
It is clearly expected that increasing the number of available channels is very effective way to improve performance of multi-channel ad hoc networks. We show in Fig. 2.8 the average aggregate throughput when six channels are used. We find that the maximum



(a) 16-node topology.

(b) 36-node topology.

Figure 2.7: Average aggregate throughput vs. packet arrival rate in three-channel single collision domain.

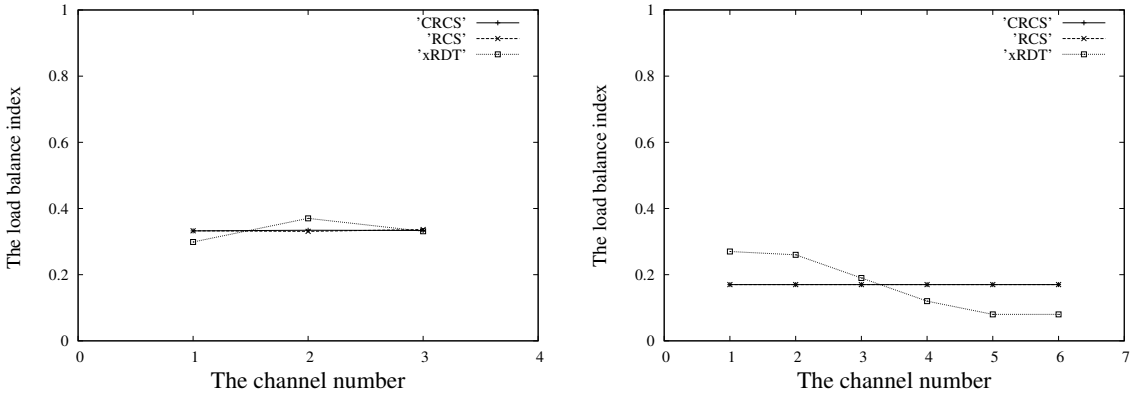


(a) 16-node topology.

(b) 36-node topology.

Figure 2.8: Average aggregate throughput vs. packet arrival rate in six-channel single collision domain.

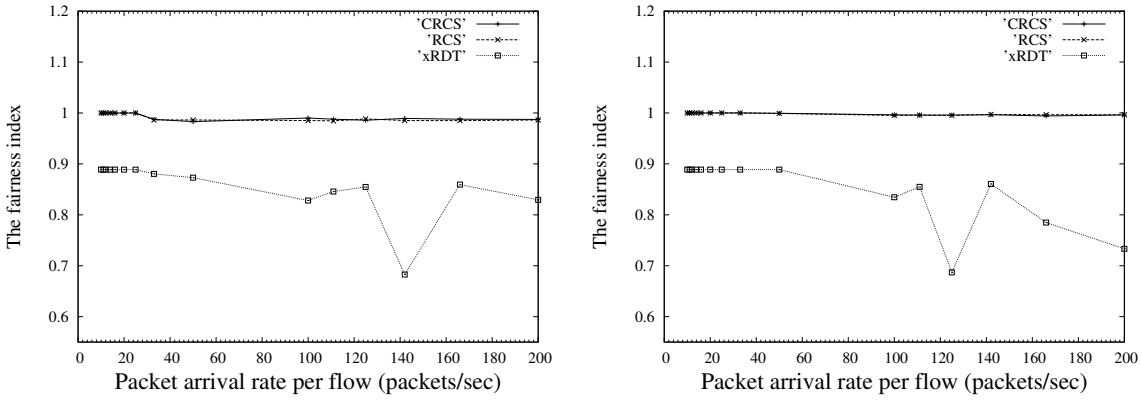
achievable performance in six channels is greater than that in three channels for xRDT. We also find that the average aggregate throughput of CCS reduces seriously even the available channels are doubled. Clearly, always fixing one channel for data transmission cannot help to improve the performance for multi-channel ad hoc networks. In contrast, our channel selection algorithm in CRCS can effectively utilize all of available channels. It can be observed that CRCS maintains the second best and almost the same performance of xRDT. In case of 36-node topology, we find that CRCS gains the same aggregate average throughput with xRDT.



(a) 36-node topology with three channels.

(b) 36-node topology with six channels.

Figure 2.9: The load balance index in single collision domain.



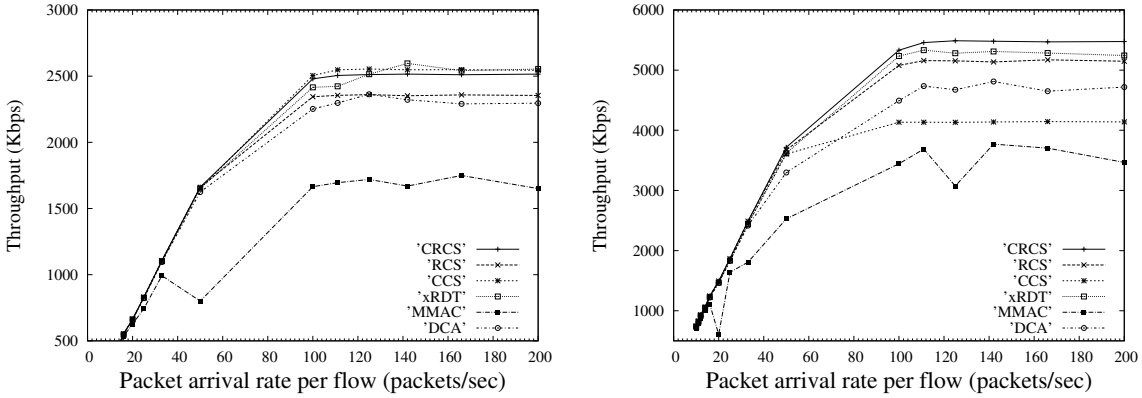
(a) 36-node topology with three channels.

(b) 36-node topology with six channels.

Figure 2.10: The Jain's fairness index vs. packet arrival rate in single collision domain.

Figure 2.9 depicts the load balance index of CRCS, RCS and xRDT in the 36-node topology. When the number of channels is three, xRDT keeps the load balance of all of channels but CRCS and RCS can achieve almost the best load balance index of channel utilization. When the number of channels is six, however, xRDT cannot keep balanced channel utilization. If we define the variation of the load balance index as the highest load balance index minus the lowest load balance index, xRDT is about 10%. On the other hand, although CRCS is a simpler channel selection algorithm than xRDT, it can keep the load balance of all of channels almost perfectly even the number of channels is six.

Furthermore, we show the fairness index of CRCS, RCS, and xRDT in Fig. 2.10. CRCS and RCS have the fairness index that is close to one, and keep the quite good performance



(a) 16-node topology.

(b) 36-node topology.

Figure 2.11: Average aggregate throughput vs. packet arrival rate in three-channel multiple collision domain.

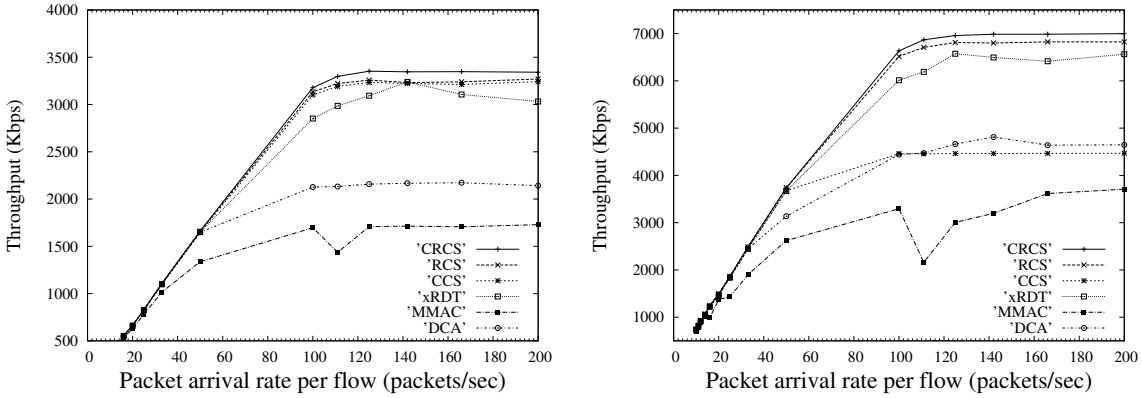
for all packet arrival rates. On the other hand, xRDT cannot keep the fairness index in the high level as CRCS and RCS.

By investigating some cases of xRDT simulations, we find that two connections cannot make data transmission. Because xRDT uses the lowest load channel selection algorithm, which seeks to the maximal throughput only for some connections, the throughput performance may become biased. In contrast, CRCS (and RCS) can achieve the quite fair throughput among all connections because of randomized channel selection.

2.3.3 Multiple collision domain

In the real ad hoc networks, we cannot always assume that all of nodes only in one transmission range. Now, we look at simulation results in the multiple collision domain.

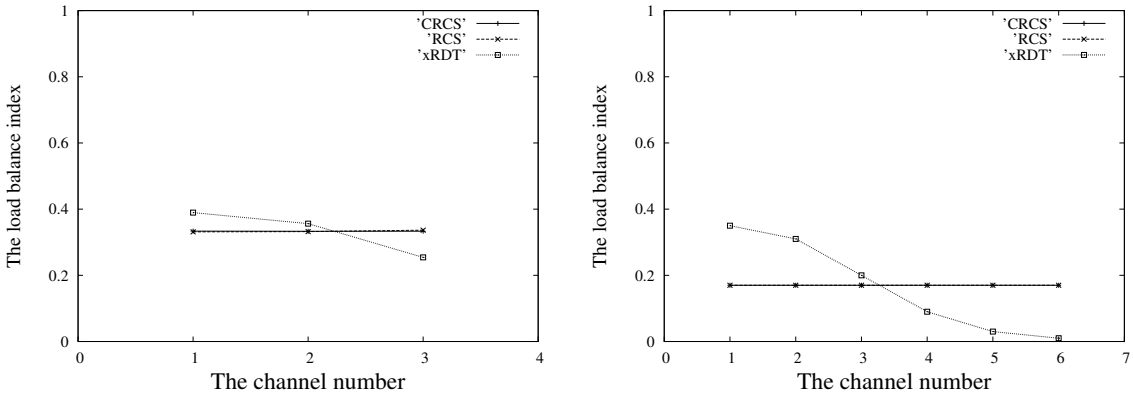
Figure 2.11 depicts the average aggregate throughput of different multi-channel MAC protocols in 16-node and 36-node topologies versus input packet arrival rate. The total of three and six channels are used in those simulations as in case of single collision domain. We note that MMAC gains the worst average aggregate throughput among all of multi-channel MAC protocols. This is because MMAC gets too much penalty on overhead of negotiating channel on the common channel. We also note that CRCS becomes more effective in multiple collision domain than single collision domain, although it is just a simple channel selection algorithm. Actually, as shown in Fig. 2.11(b), CRCS can get a higher average aggregate throughput than xRDT. The potential threat of the channel selection algorithm in xRDT is that it cannot always find accurately a good channel for the other connections in multiple collision domain. By *good channel*, we mean the channel that a node can reduce



(a) 16-node topology.

(b) 36-node topology.

Figure 2.12: Average aggregate throughput vs. packet arrival rate in six-channel multiple collision domain.

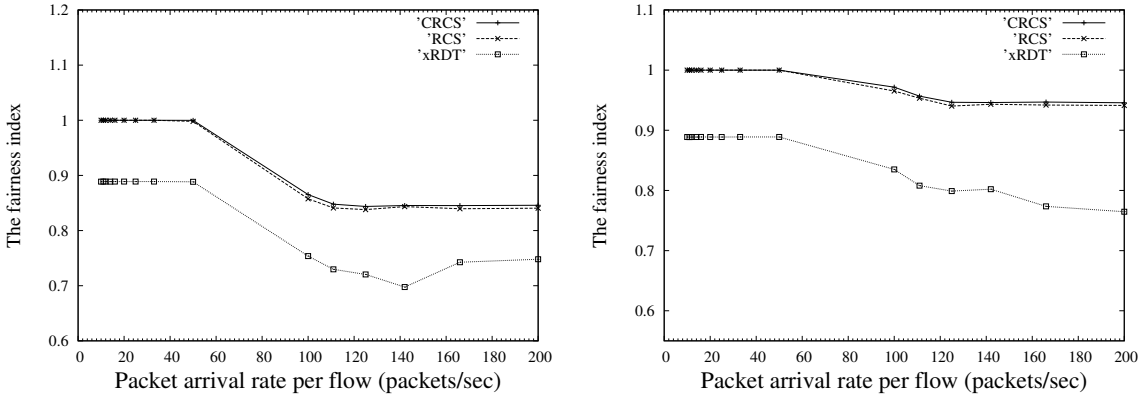


(a) 36-node topology with three channels.

(b) 36-node topology with six channels.

Figure 2.13: The load balance index in multiple collision domain.

collisions between the other nodes. Since all of nodes are not in the same transmission range in multiple collision domain, nodes cannot overhear the information of channel on the other nodes. That makes nodes impossible to find always a good channel for the other nodes. Therefore, xRDT starts to reduce its throughput. Such potential threat in xRDT deteriorates the performance by increasing the number of channels. As shown in Fig. 2.12, when the number of channels increases, the average aggregate throughput of xRDT drops obviously comparing to xRDT in the single collision domain. On the other hand, CRCS keeps effective to improve its throughput, and always outperforms against xRDT with respect to the average aggregate throughput in both 16-node and 36-node topologies.



(a) 36-node topology with three channels.

(b) 36-node topology with six channels.

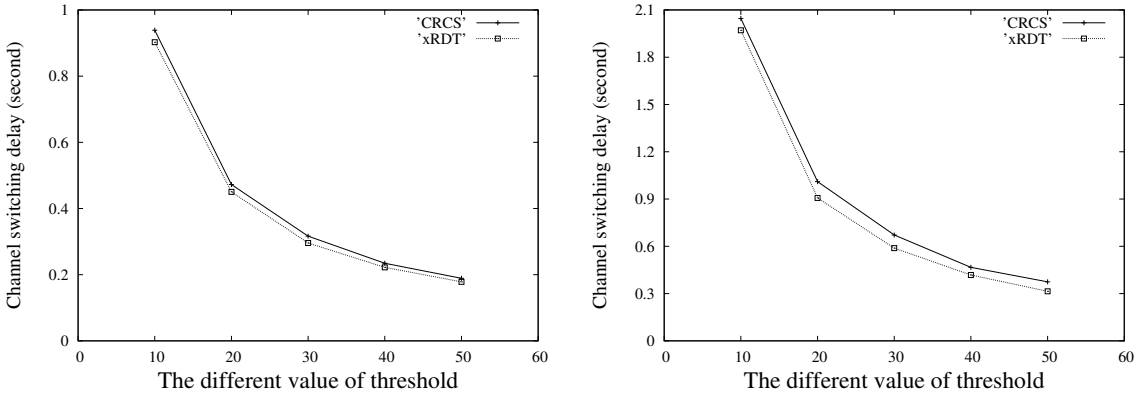
Figure 2.14: The Jain's fairness index vs. packet arrival rate in multiple collision domain.

We also show the load balance index of all channels in Fig. 2.13 and the fairness index in Fig. 2.14. Compared to Fig. 2.9, we note that xRDT has more unstable load balance index in both cases of three and six channels. In fact, the variation of the load balance index on xRDT is over 30% in the six-channel case. Meanwhile, CRCS and RCS keep almost the best load balance index in multiple collision domain. Like the single collision domain, CRCS and RCS have the good fairness index that is close to one for small value of packet arrival rates. Although the high fairness cannot be kept as increasing packet arrival rate, however, CRCS outperforms xRDT.

In summary, CRCS has some advantages compared to xRDT.

- Simple. To find the lowest load in xRDT, each node should calculate the longest idle time of channels. This process will produce the cost or expense power on each node. For example, if the connection keeps a long time, such as an hour or one day, xRDT will pay the big cost for consuming the power. On the other hand, the channel is changed randomly in CRCS. We believe that CRCS is more efficient to save the power and is more scalable.

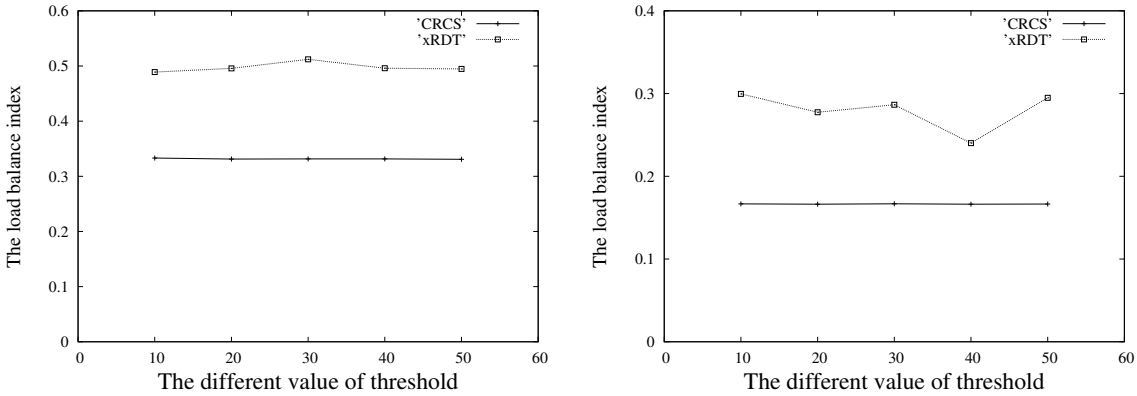
- Effective. When the number of channels, nodes and connections are large in multiple collision domain, to find the lowest load channel in xRDT is not an effective way to improve the performance of multi-channel ad hoc networks. On the other hand, CRCS, which uses a simple randomized channel selection, can not only increase the performance of ad hoc networks but also keep the load balance of all channels.



(a) Three-channel case.

(b) Six-channel case

Figure 2.15: Channel switching delay vs. the different value of threshold in 36-node multiple collision domain. Packet arrival rate per flow is 100 (packets/sec).



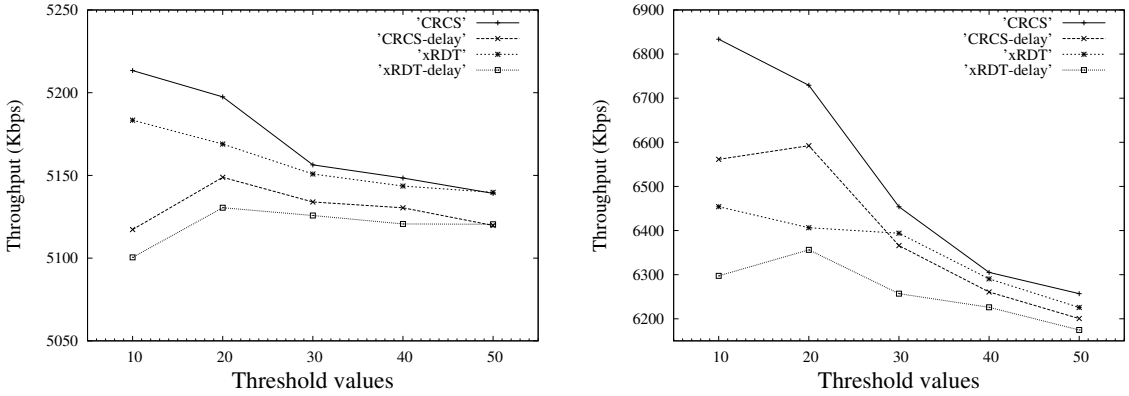
(a) Three-channel case.

(b) Six-channel case

Figure 2.16: The load balance index vs. the different value of threshold in 36-node multiple collision domain. Packet arrival rate per flow is 100 (packets/sec).

2.3.4 Effects of the threshold

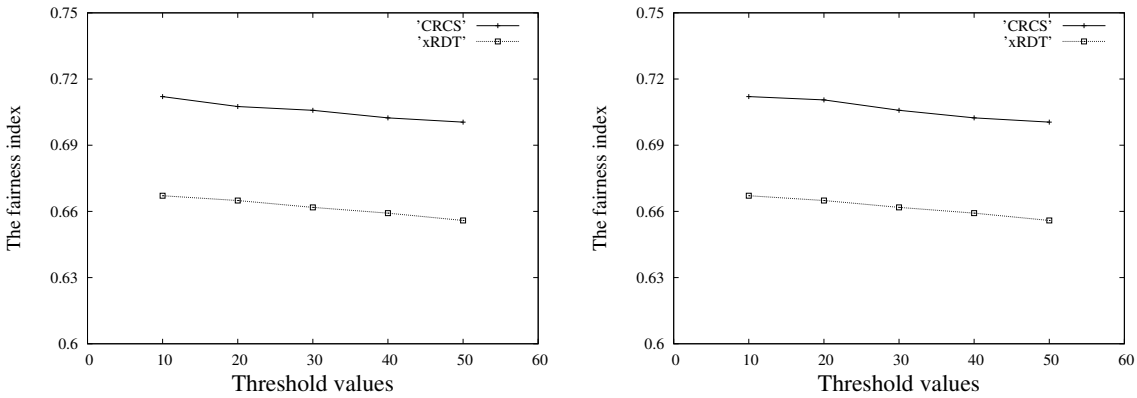
So far, we have assumed a scenario that xRDT and xRDT-based protocols, including CRCS, change the channel packet-by-packet in their channel selection algorithms. But in real ad hoc networks, changing the channel by every packet may produce large amount of channel switching delay because the network interface must change a channel for every packet transmission. To reduce the switching delay, we could set the threshold that is bigger than one packet. We explore the effects of the threshold on the performance in xRDT and CRCS.



(a) Three-channel case.

(b) Six-channel case.

Figure 2.17: Throughput vs. the different value of threshold in 36-node multiple collision domain. Packet arrival rate per flow is 100 (packets/sec) .



(a) Three-channel case.

(b) Six-channel case

Figure 2.18: The fairness index vs. the different value of threshold in 36-node multiple collision domain. Packet arrival rate per flow is 100 (packets/sec)

Figure 2.15 depicts channel switching delay versus the threshold with 36-node multiple collision domain for three-channel case 2.15(a) and six-channel case 2.15(b). From Fig. 2.15, it is observed that the channel switching delay reduces as increasing the threshold. This is because the large threshold can reduce the switching time of channels. Although we find out that CRCS can give a little higher channel switching delay than xRDT, xRDT gives the high cost on the load balance index. As shown in Fig. 4.6, CRCS has the same probability to change the different channel over the threshold, while xRDT prefers to use one channel.

Therefore, CRCS produces more chance to switch its channel comparing to xRDT. In other words, CRCS has high switching rate as compared to xRDT. Note that, although we do not show the load balance index of all channels, xRDT and CRCS have the similar performance as shown in Fig. 2.13. The case where the number of channels is six in 36-node multiple domain collision is also investigated, as shown in Fig. 2.16(b), that CRCS also provides a good load balance index.

In Fig. 2.17(a), it is observed that the different threshold has the different throughput. We also find out that CRCS can provide higher throughput than xRDT when the threshold is small. Since the number of changing channels is increased at the small threshold, xRDT has a higher possibility to find out an *ungood channel* and therefore xRDT cannot provide high throughput. When the threshold is large, CRCS and xRDT provide lower throughput. This is because reducing the number of changing channels increases the possibility of the collision. It is observed that from Fig. 2.17(b) CRCS can provide much higher throughput than xRDT when the threshold is small. We also calculate the throughput of CRCS and xRDT including channel switching delay. As expected the throughput is reduced by including channel switching delay. However, CRCS still can provide higher throughput than xRDT.

Figure 2.18 depicts the fairness index versus the threshold with 36-node multiple collision domain for three-channel case 2.18(a) and six-channel case 2.18(b). It is observed that the fairness index slightly reduces, but almost is insensitive, as increasing the threshold and the fairness index of CRCS is better than xRDT at different thresholds.

It is clear that the higher throughput can be given at the small threshold. However, the channel switching delay is also increased. How to find the optimal threshold to balance the throughput and the load balance of all channels is still an interesting open issue. It is obvious that the optimal threshold depends on the network situation, such as the number of nodes, channels and connections. In the real ad hoc networks, the number of nodes and connections are often dynamically changed, so it is desirable to make that the threshold is adapted to the network situation. Changing the threshold dynamically to achieve the high throughput and the low switch delay is left as a future work.

2.4 Conclusion

In this paper, we proposed a channel selection algorithm CRCS based on xRDT. The core idea of CRCS is using the active channel for data transmission until the amount of data packets reaches a threshold, and then changing to one of the available channels other than the active channel. CRCS solves the problems of which channel and how long the channel should be used by a pair of nodes. By using ns-2 network simulator, we found that CRCS is simpler and effective to increase the capacity of ad hoc networks and keep the load balance of all channels.

Chapter 3

Using Game Theory to Investigate Channel Selection Algorithm

It is obvious that using multi-channel MAC protocol can improve the performance of ad hoc networks. At the same time, it also is clear that the strategy for selecting the channel can affect to increase the performance of ad hoc networks. Therefore, the channel selection strategy is one of the important issues for the multi-channel MAC protocol in ad hoc networks. Selecting the channel stochastically is a basic strategy in multi-channel MAC protocol using *single interface*, because wireless nodes cannot hear communication taking place on a different channel. In this chapter, we propose a multi-channel MAC protocol using single interface. Then, we consider two kinds of categories of stochastic channel selection strategies for the multi-channel MAC protocol using single interface. One category consists of strategies that select a new channel from all available channels. The other category is the set of strategies that select a new channel from all available channels *except the active channel*. By using game theory, we investigate and show that wireless node using these two categories can operate the stable point if a new channel is selected with *equal probability*. Furthermore, we argue that the higher expected payoff can be obtained if a new channel is selected other than the active channel. By computer simulation, we show that our assertion is effective.

Note: The materials of this chapter have been appeared in [37].

3.1 Background

There has been a considerable amount of research on channel allocation in the infrastructure based mode for WLANs [38]. There are two major categories of channel allocation that are always used in WLANs: centrally managed networks and uncoordinated networks. A typical metric for these two categories is the interference.

In centrally managed networks, there exists a central entity that decides and assigns channels to each access point (AP) such that the performance metric of the network is optimized. In [39], graph coloring technique provides an efficient way to solve the problem of minimal interference among all of available channels. On the other hand, in uncoordinated networks, the main feature is the distributed execution in which channel is allocated in a distributive way by each AP instead of a central controller. This concept is attractive, since the network performance is improved by adjusting each AP alone. In [40], the objective is to assign overlapping channels to APs in such a way that minimizes the total weighted interference by each AP. Such interference is weighted by the transmitting power of AP as the interference factor. Each AP has two algorithms to minimize such interference. One is to pick randomly a channel for allocation and the other one is simply to pick the first channel of the ascending ordered channel list.

The channel allocation has been also researched on ad hoc networks. It is clear that ad hoc networks belong to uncoordinated networks, since there does not exist one central node who can control all of other nodes in ad hoc networks. Meanwhile, game theory provides a straightforward tool to study the channel allocation problem on ad hoc networks. Actually, we can find a good strategy to provide the higher performance for ad hoc networks by using game theory. In [41] and [42], the authors analyze the channel allocation strategies using the non-cooperative channel allocation game and propose several algorithms that enable wireless nodes to converge to Nash equilibrium. However, their results can be only applied to single-hop networks. In [43], the authors extend the channel allocation game to two-hop wireless network. Furthermore, in [44], the authors extend the channel allocation game to arbitrary hops using cooperative game. Unfortunately, all of their models do not apply to multi-channel MAC protocol using the single interface and their algorithms are not dynamic for wireless nodes. In the other words, their approaches are not for the channel selection strategy. In this chapter, we use the game approach to provide the simple, effective and dynamic channel selection strategy for multi-channel MAC protocol using the single interface.

3.2 System model

In our model, we assume that the available frequency band is divided into m orthogonal channels of the same bandwidth in our multi-channel MAC protocol. We denote the set of available orthogonal channels by $CH = \{ch_1, ch_2, \ldots, ch_m\}$. We also assume that there exists L communication links. Each communication link has two wireless nodes, one is a sender and the other one is a receiver. We also assume that each wireless node is equipped with a switchable interface. The node can either receive or transmit packets at a time but not both simultaneously. However, nodes can simultaneously sense carriers on all channels when they are idle. Figure 3.1 presents an example of three communication links using three different channels.

In our model, we assume that the receiver is the decision maker to decide which channel can be used. Before transmitting, the receiver randomly selects one channel from CH and puts his interface on that channel. Note that only the receiver himself can know the information of which channel is used and the other receivers do not know this information. We also assume that at least there exists two receivers who reside in one interference range, which means that each communication link will interfere the transmission of the other communication links if they are using the same channel.

We assume that all of communication links use the same data transmission rate r and all of data packets have the same size p. Thus, the transmission time of data packet is p/r. We choose p/r as the unit of time that is equal to one. We also assume that each communication link needs to spend the time c before sending its data packet. The time c includes channel switching delay and the time consumes on exchanging control packets, such as RTS/CTS/ACK. Note that c is not a constant time, since the different time is consumed by back-off mechanism due to the collision of packets. For instance, in our model, if there are more than three communication links to make data transmission in one interference range, some of communication links will compete one channel that causes to consume more time on c.

In Fig. 3.2, we show the transmitting flow of data packets in our multi-channel MAC protocol. As shown in this figure, the transmitter broadcasts RTS to the receiver. If the transmitter did not receive CTS from the receiver, it switches to the next channel to broadcast RTS. By next channel, we assume that the channel is selected by a cyclic order of the number of current channel. When the transmitter gets CTS from the receiver on the channel, the transmitter uses this channel to send the data packet. After the receiver receives *one data packet*, the receiver decides to change a new channel to make data transmission. After data transmission, the receiver broadcasts ACK to the transmitter.

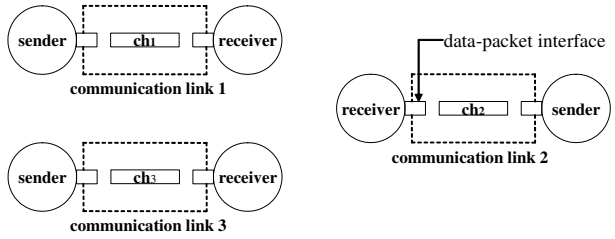


Figure 3.1: An example of three communication links.

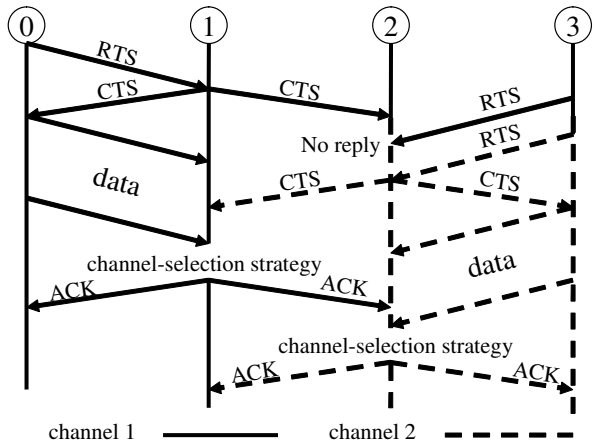


Figure 3.2: In this topology, the distance between neighboring nodes is 200m, the transmission range is 250m and the carrier sensing range is 550m.

3.3 Game formulation

We refer to the receiver as a selfish player who is rational and intelligent in the communication link. Let us denote by I the set of the players. For example, we have $I = \{1, 2\}$ in the case of Fig. 3.2.

We assume that channel selection consists of initial and strategic phases in which each player decides the channel to be used. In initial phase, each player independently selects one channel from all of available channels CH with some probability for the first time transmission. Then, the player can use two different categories of the stochastic channel selection strategies in strategic phase. We summarize two categories as follows:

- Category 1 (C_1), the player selects one channel from all available channels except the active channel with some probability.

- Category 2 (C_2), the player selects one channel from all available channels with some probability.

For the two categories C_1 and C_2, we define two channel selection games, one using C_1 and the other using C_2 in strategic phase.

In each game, we denote by S_i the set of pure strategies for $i \in I$. Then, we can specify a pure strategy of the player i by $s_{i,ch,ch'}$, where $ch \in CH$ denotes the channel selected in initial phase and $ch' \in CH$ denotes the channel selected in strategic phase. Note that we set $ch \neq ch'$ in C_1 but not in C_2. Then, the set of pure strategies S_i is $\{s_{i,ch_1,ch_2}, s_{i,ch_1,ch_3}, \ldots, s_{i,ch_m,ch_{m-1}}\}$ if we use C_1 or $\{s_{i,ch_1,ch_1}, s_{i,ch_1,ch_2}, \ldots, s_{i,ch_m,ch_m}\}$ if we use C_2.

Suppose that each player sends one data packet of size p in the communication link with data transmission rate r. Then, we assume that the player enjoys a reward $1(= p/r)$ in our unit of time if the data packet can be received using *successful channel*. By the successful channel, we mean that it cannot cause the collision with the other channels. Because each player must spend the amount of time c for data transmission, we assume that the payoff u_+ for successful channel is given by

$$u_+ = +1 - c, \tag{3.1}$$

if the player uses the successful channel. On the other hand, if the player uses *unsuccessful channels*, in which the player cannot receive data packets successfully, we assume that the payoff u_- for unsuccessful channel is given by

$$u_- = -1 - c. \tag{3.2}$$

In Table 3.1, we present the example that is all of possible payoffs of pure strategies in our stochastic channel selection game using C_1 with two players and three channels. For example, if player 1 uses s_{1,ch_1,ch_2} and player 2 uses s_{2,ch_2,ch_1}, then the payoff is $2 - 2c$ for players 1 and 2. This is because two players get $u_+ = +1 - c$ at the first time data transmission on successful channel and also get $u_+ = +1 - c$ at the second time.

Let us denote by Σ_i the set of probability distributions over S_i. Then, the pure strategies will be probabilistically determined by a mixed strategy $\sigma_i \in \Sigma_i$ which is a probability distribution over S_i. The probability that player i selects a particular pure strategy $s_i \in S_i$ is specified by $\sigma_i(s_i)$. For the simplicity of presentation, we denote by $s \in S$ a pure-strategy profile, where $S = S_1 \times S_2 \times \cdots \times S_n$. Similarly, we denote by $\sigma \in \Sigma$ a mixed-strategy profile that is realized when all the players specify their individual mixed strategy, where $\Sigma = \Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_n$. When we focus on strategies of all players except player i, we write player i's deleted pure-strategy profile s_{-i} and deleted mixed-strategy profile σ_{-i}. We denote by S_{-i} and Σ_{-i} the set of player i's deleted pure-strategy and mixed-strategy profiles, respectively.

Table 3.1: In this form, we present that the payoff of players 1 and 2 select their pure strategies, where player 1 is the row player and player 2 is the column player. In each cell, the first value is the payoff of player 1, whereas the second is the payoff of player 2.

player 1 \ player 2		*ch*1		*ch*2		*ch*3	
		*ch*2	*ch*3	*ch*1	*ch*3	*ch*1	*ch*2
*ch*1	*ch*2	(-2-2*c*,-2-2*c*)	(-2*c*,-2*c*)	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)	(-2*c*,-2*c*)
	*ch*3	(-2*c*,-2*c*)	(-2-2*c*,-2-2*c*)	(2-2*c*,2-2*c*)	(-2*c*,-2*c*)	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)
*ch*2	*ch*1	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)	(-2-2*c*,-2-2*c*)	(-2*c*,-2*c*)	(-2*c*,-2*c*)	(2-2*c*,2-2*c*)
	*ch*3	(2-2*c*,2-2*c*)	(-2*c*,-2*c*)	(-2*c*,-2*c*)	(-2-2*c*,-2-2*c*)	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)
*ch*3	*ch*1	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)	(-2*c*,-2*c*)	(2-2*c*,2-2*c*)	(-2-2*c*,-2-2*c*)	(-2*c*,-2*c*)
	*ch*2	(-2*c*,-2*c*)	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)	(2-2*c*,2-2*c*)	(-2*c*,-2*c*)	(-2-2*c*,-2-2*c*)

Each player i receives a payoff which depends on $s \in S$, viz, $u_i(s)$. The expected payoff $\bar{u}_i(\sigma)$ of player i when the players all participate in a mixed-strategy profile $\sigma \in \Sigma$ is

$$\bar{u}_i(\sigma) = \sum_{s \in S} \left(\prod_{j \in I} \sigma_j(s_j) \right) u_i(s). \tag{3.3}$$

As a special case of Eq. (3.3), we can show that player i's expected payoff when he plays the pure strategy $s_i \in S_i$ against the deleted mixed-strategy profile $\sigma_{-i} \in \Sigma_{-i}$ is

$$\bar{u}_i(s_i, \sigma_{-i}) = \sum_{s_{-i} \in S_{-i}} \left(\prod_{j \in I \setminus \{i\}} \sigma_j(s_j) \right) u_i(s_i, s_{-i}). \tag{3.4}$$

Definition 1 (Nash equilibrium) *The pure-strategy profile $s^* \in S$ or a* mixed-strategy *profile $\sigma^* \in \Sigma$ defines a Nash equilibrium, if for every player i, we have*

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*), \tag{3.5}$$

or

$$\bar{u}_i(\sigma_i^*, \sigma_{-i}^*) \geq \bar{u}_i(\sigma_i, \sigma_{-i}^*), \tag{3.6}$$

where $\forall i \in I$ and $\forall s_i \in S_i$ or $\forall \sigma_i \in \Sigma_i$, and s_i^* or σ_i^* maximizes the objective function $u_i(s_i, s_{-i})$ or $\bar{u}_i(\sigma_i, \sigma_{-i})$ for the given deleted strategy profile $s_{-i} \in S_{-i}$ or $\sigma_{-i} \in \Sigma_{-i}$. In other words, in Nash equilibrium, no individual player can increase his payoff from unilateral deviation.

3.4 Nash equilibria for our games

In this section, we show that there exists the stable operating point to which selfish players agree, i.e., the Nash equilibria for our stochastic channel selection games.

Since each player can only transmit or listen on one channel at a time based on our multi-channel MAC protocol, it cannot always select one channel with probability one as its best strategy. Therefore, it is better to find out the mixed-strategy Nash equilibrium rather than the pure-strategy Nash equilibrium for our stochastic channel selection games. We can use the following proposition to find the mixed-strategy Nash equilibrium [45].

Proposition 1 *Let $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ be the mixed-strategy profile for the n-player game. For any player $i = 1, 2, \ldots, n$, let $\bar{u}_i(s, \sigma_{-i})$ be the expected payoff to player i when playing $s \in S_i$ against $\sigma_{-i} \in \Sigma_{-i}$. Then, σ is a Nash equilibrium iff the following two conditions hold:*

1. *If $s, s' \in S_i$ are two strategies that occur with positive probability in σ_i, then $\bar{u}_i(s, \sigma_{-i}) = \bar{u}_i(s', \sigma_{-i})$.*
2. *If $s, s' \in S_i$ are two strategies where s occurs with positive probability in σ_i and s' occurs with zero probability in σ_i, then $\bar{u}_i(s, \sigma_{-i}) \geq \bar{u}_i(s', \sigma_{-i})$.*

In general, there are infinitely many mixed-strategy profiles. However, only finitely many subsets of S can be supports of Nash equilibria. In our stochastic channel selection games, we find mixed-strategy Nash equilibria with support on S.

Definition 2 *The mixed-strategy Nash equilibrium $\sigma^* = (\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)$ is called totally mixed if every pure strategy $s_i \in S_i$ for each player i occurs with positive probability in σ_i^*.*

We want to find totally mixed-strategy Nash equilibria for our channel selection games. It is because the totally mixed-strategy Nash equilibrium can improve the load balance of all channels as shown in Chapter 2. Thus, to seek a totally mixed-strategy Nash equilibrium for two-player and three-channel stochastic channel selection game using C_1, we need to find a solution of the system of equations,

$$\bar{u}_1(s_{1,ch_1,ch_2}, \sigma_2) = \cdots = \bar{u}_1(s_{1,ch_3,ch_2}, \sigma_2), \tag{3.7}$$

$$\bar{u}_2(\sigma_1, s_{2,ch_1,ch_2}) = \cdots = \bar{u}_2(\sigma_1, s_{2,ch_3,ch_2}), \tag{3.8}$$

with the constraint

$$\sum_{ch,ch' \in CH, ch \neq ch'} \sigma_i(s_{i,ch,ch'}) = 1, \tag{3.9}$$

of unknowns $\sigma_i(s_{i,ch,ch'}) > 0$ for $i = 1, 2$.

In Eqs. (3.7) and (3.8), there are twelve different $\sigma_i(s_{i,ch,ch'})$ and twelve equations. If we set

$$\sigma_1^*(s_{1,ch,ch'}) = \sigma_2^*(s_{2,ch,ch'}) = 1/6, \tag{3.10}$$

for $\forall ch \in CH$ and $\forall ch' \in CH$ with $ch \neq ch'$, then we can confirm that $\sigma^* = (\sigma_1^*, \sigma_2^*)$ is a totally mixed-strategy Nash equilibrium for our stochastic channel selection game using C_1 with two players and three channels.

In n-player and m-channel stochastic channel selection game using C_1, if we set

$$\sigma_i^*(s_{i,ch,ch'}) = 1/[m \cdot (m-1)], \tag{3.11}$$

for $i \in \{1, 2, \ldots, n\}$, $\forall ch \in CH$ and $\forall ch' \in CH$ with $ch \neq ch'$, then we can confirm that

$$\bar{u}_i(s_{i,ch_1,ch_2}, \sigma_{-i}^*) = \cdots = \bar{u}_i(s_{i,ch_m,ch_{m-1}}, \sigma_{-i}^*), \tag{3.12}$$

for $i \in \{1, 2, \ldots, n\}$ in n-player and m-channel stochastic channel selection game using C_1. See Appendix C for the proof of Eq. (3.12). We then obtain the following proposition.

Proposition 2 *Let us consider $\sigma^* = (\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)$, where $\sigma_i^*(s_{i,ch,ch'})$ is given by Eq.(3.11). Then, σ^* is a totally mixed-strategy Nash equilibrium for our stochastic channel selection game using C_1 with n players and m channels.*

Note that, in our stochastic channel selection game using C_1, the joint probability $\sigma_i^*(s_{i,ch,ch'})$ can be rewritten as

$$\sigma_i^*(s_{i,ch,ch'}) = \rho(ch'|ch, i) \times \rho(ch, i),$$

where $\rho(ch'|ch, i)$ is the conditional probability of selecting ch' in strategic phase, given that player i selects ch in initial phase, and $\rho(ch, i)$ is the probability of player i selecting ch in initial phase. In our model, we assume that the player randomly selects one channel from all of available channels with equal probability. Thus, we have $\rho(ch, i) = 1/m$. Then, we need to set $\rho(ch'|ch, i) = 1/(m-1)$. In other words, if we assume that each player independently and randomly selects one channel with equal probability for the first time transmission, then each player selects a new channel from all available channel except the active channel *with equal probability*. Therefore, σ^* is a totally mixed-strategy Nash equilibrium, when each player can select a new channel from available channels *except the active channel* with the same equal probability after independently and randomly selecting one channel from all of available channels with equal probability for the first time transmission. We call this channel selection strategy as Conditionally Randomized Channel Selection (CRCS).

Similarly, for n-player and m-channel stochastic channel selection game using C_2, if we set

$$\sigma_i^*(s_{i,ch,ch'}) = 1/m^2, \tag{3.13}$$

for $i \in \{1, 2, \ldots, n\}$, $\forall ch \in CH$ and $\forall ch' \in CH$ in n-player and m-channel stochastic channel selection game using C_2, then we can confirm that

$$\bar{u}_i(s_{i,ch_1,ch_1}, \sigma_{-i}^*) = \cdots = \bar{u}_i(s_{i,ch_m,ch_m}, \sigma_{-i}^*), \tag{3.14}$$

for $i \in \{1, 2, \ldots, n\}$ in n-player and m-channel stochastic channel selection game using C_2. Also, see Appendix C for the proof of Eq. (3.14). Therefore, after selecting one channel from all of available channels with equal probability for the first time transmission, selecting a new channel from all of available channels with equal probability again leads to the totally mixed-strategy Nash equilibrium. We call this channel selection strategy as Random Channel Selection (RCS).

3.4.1 The expected payoff in repeated game

We have shown that CRCS and RCS are Nash equilibria for our stochastic channel selection games using C_1 and C_2. However, to understand a long-term relationship among players, we need the repeated game. Essentially, we consider that our stochastic channel selection games in repeated model are situations in which players repeatedly engage in the strategic phases using C_1 and C_2 infinitely. Then, the total expected payoff Q_i of player i is given by

$$Q_i = \sum_{t=1}^{\infty} \delta^t q_i(t),$$

where $q_i(t)$ is the expected payoff of player i in the t-th strategic phase for $t = 1, 2, \ldots$, and $\delta \in (0, 1)$ is a discount factor of each player for the future payoff.

Since we assume players are rational and intelligent, the player will use CRCS and RCS in every strategic phase for our stochastic channel selection games using C_1 and C_2. It can be shown that the expected payoff $r_i(t)$ in both CRCS and RCS with n players using m channels is computed as $4((m-1)/m)^{n-1} - 2 - 2c$ (see Appendix C) for every strategic phase. Therefore, the total expected payoff is obtained by

$$Q_i = \left(4\left(\frac{m-1}{m}\right)^{n-1} - 2 - 2c\right) / (1 - \delta),$$

for each player using both CRCS and RCS.

We, however, should consider that CRCS and RCS have different values of cost c. More precisely, we can see that the cost of CRCS is lower than that of RCS. A rationale is based on the probability p that packets collide in IEEE 802.11 DCF under N wireless nodes sharing a single channel [46]

$$p = 1 - (1 - \tau)^{N-1}, \tag{3.15}$$

where τ is the probability that a wireless node transmits a packet in a time slot. Since CRCS avoids selecting the active channel as a new channel, we consider that the number of wireless nodes sharing the same channel in CRCS is smaller than in RCS, which implies that CRCS has the lower collision probability than RCS. Therefore, the expected time on consuming back-off mechanism in CRCS is shorter than RCS. That means CRCS pays the

lower cost than RCS. Later, we will show the packet loss rate by computer simulation to support our assertion.

3.4.2 Summary

Efficiency and *equity* are two main advantages for CRCS and RCS. CRCS and RCS possess efficiency because CRCS and RCS not only keep the load balance of all channels but also increase the performance of ad hoc networks. The property of the load balancing of CRCS and RCS has been shown in the last chapter. We will use computer simulation to show the second property that CRCS and RCS achieve the higher performance than the other channel selection distribution. CRCS and RCS are also equity because each player using CRCS and RCS can get the same payoff to each other. In other words, if one player were to deviate from CRCS and RCS, then the other player(s) would no longer have the best selection. These two advantages are reasons why each player agrees to use CRCS and RCS in our stochastic channel selection games. The difference between CRCS and RCS is that CRCS gives the higher expected payoff than RCS. Therefore, we believe that can help CRCS provide the better performance than RCS. We will also use computer simulation to show this result.

3.5 Channel selection algorithm

In this section, we design the channel selection algorithm that helps wireless node behave the stable operating point.

This algorithm is based on our stochastic channel selection strategies. The basic idea is that each receiver independently and randomly selects one channel all of available channels with equal probability in initial phase. A new channel is selected from available channels by using CRCS or RCS. The channel is updated every time when the receiver received one data packet. The receiver then informs the new channel to the neighboring nodes by using broadcast through the old channel. After informing the new channel, the receiver begins to listen on the new channel. Our channel selection algorithm is summarized as follow.

```
initialize the receiver's channel (e.g., channel ch)
loop
  if one packet received then
    select the new channel ch' by using CRCS or RCS
    broadcast the new channel ch'
    switch the interface to the new channel ch'
    ch = ch'
```

```
    end if
end loop
```

3.6 Performance evaluation

In this section, we evaluate the performance of stochastic channel selection strategies by network simulator ns-2 [35]. In network simulator, we use IEEE 802.11b protocol that the available frequency band is divided into three channels orthogonal channels of the same bandwidth using the frequency division multiple access (FDMA) method. We compare the proposed channel selection strategies CRCS with RCS and the other stochastic channel selection strategies, Conditional Channel Selection (CCS) and Channel Selection (CS). Please noted that Conditional Channel Selection is a different channel selection algorithm with Constant Channel Selection (CCS) that we have used in Chapter 2. After independently and randomly selecting one channel from all of available channels with equal probability for the first time transmission, each player i (receiver of pair of source and destination nodes) selects a new channel according to the mixed-strategy

$$\sigma_i(s_{i,ch_1,ch_2}) = 0.4/3, \qquad \sigma_i(s_{i,ch_1,ch_3}) = 0.6/3,$$

$$\sigma_i(s_{i,ch_2,ch_1}) = 0.4/3, \qquad \sigma_i(s_{i,ch_2,ch_3}) = 0.6/3,$$

$$\sigma_i(s_{i,ch_3,ch_1}) = 0.4/3, \qquad \sigma_i(s_{i,ch_3,ch_2}) = 0.6/3,$$

in Constant Channel Selection (CCS), and for $\forall ch \in CH$

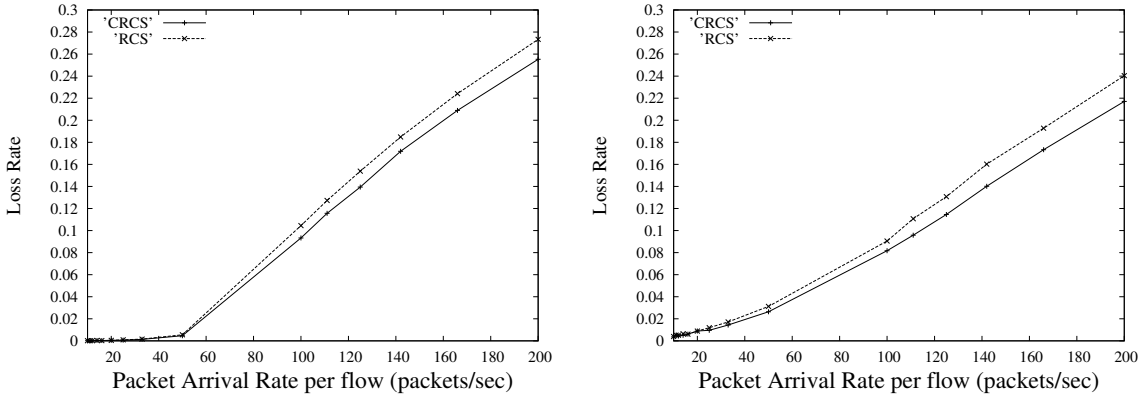
$$\sigma_i(s_{i,ch,ch_1}) = 0.2/3,$$

$$\sigma_i(s_{i,ch,ch_2}) = 0.4/3,$$

$$\sigma_i(s_{i,ch,ch_3}) = 0.4/3,$$

in CS.

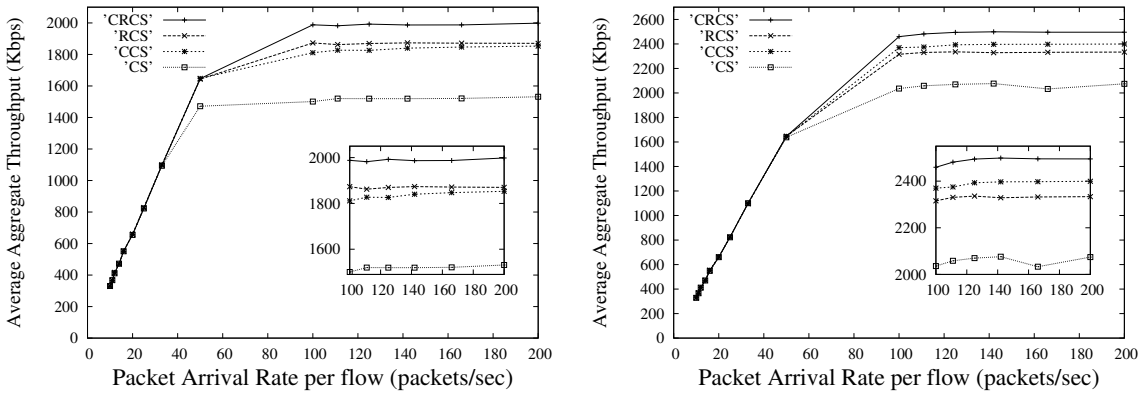
We assume the following common parameters in each simulation; all the radio parameters are ns-2 defaults; the radio propagation mode is the two-ray ground model with transmission range $250m$; carrier sensing and interference ranges are both $550m$; the number of nodes is 16 and 36 nodes; the distance of two nodes is $140m$ and $200m$. We choose half of nodes as transmitters and half of nodes as receivers. Each source node generates and transmits constant-bit rate (CBR) traffic to each destination node. For each simulation, we examine the average aggregate throughput and average packet delay.



(a) The distance of two nodes is $140m$.

(b) The distance of two nodes is $200m$.

Figure 3.3: Loss rate vs. packet arrival rate in 16-node and three-channel topology.



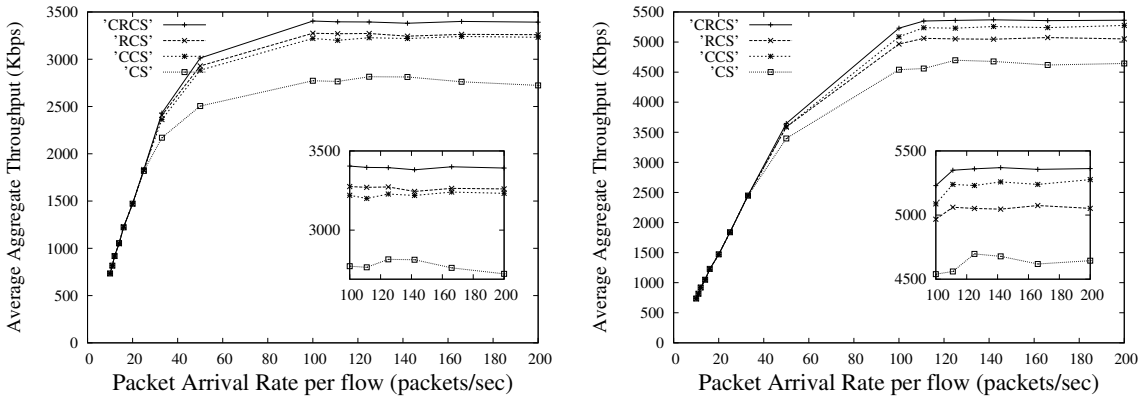
(a) The distance of two nodes is $140m$.

(b) The distance of two nodes is $200m$.

Figure 3.4: Average aggregate throughput vs. packet arrival rate in 16-node and three-channel topology

In Fig. 3.3, we show the packet loss rate of CRCS and RCS versus the packet arrival rate in 16-node topology with three channels. We can clearly observe that CRCS gives lower packet loss rate than RCS. Therefore, it can be said that the cost of CRCS is lower than the cost of RCS.

Figures 3.4 and 3.5 show the average aggregate throughput of different stochastic channel selection strategies in 16-node and 36-node topologies as increasing input packet arrival rate, when three channels are used. As shown in these figures, all of the different stochastic channel selection strategies using our multi-channel MAC protocol yield the similar performance until the arrival rate is 40 packets/sec, because the load is too low to exploit



(a) The distance of two nodes is $140m$.

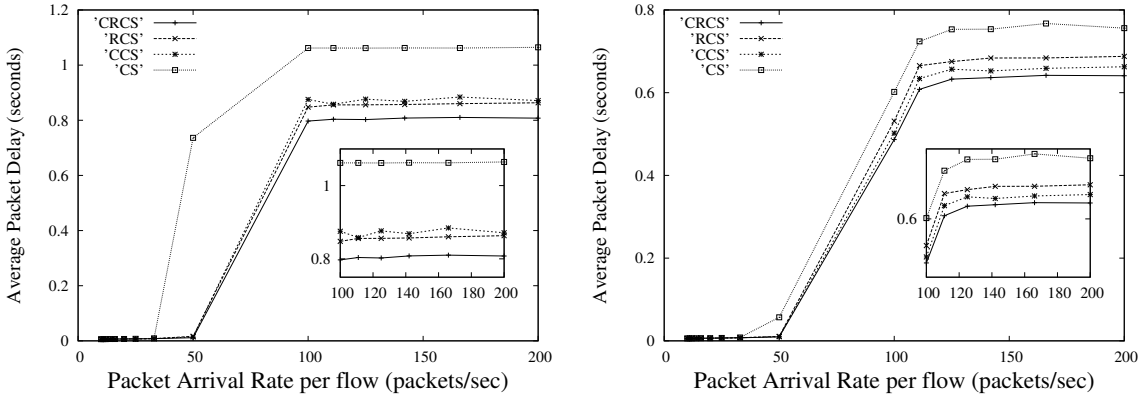
(b) The distance of two nodes is $200m$.

Figure 3.5: Average aggregate throughput vs. packet arrival rate in 36-node and three-channel topology

the multiple data channels. If we increase the arrival rate more than 40 packets/sec, then we can observe that the strategies of selecting multiple channels significantly influence the average aggregate throughput.

It is clearly expected, and also observed that CRCS and RCS are more effective to increase the throughput than CCS and CS, respectively. This is because CRCS and RCS can converge to Nash equilibria for our stochastic channel selection games. Each player at Nash equilibria can always select one channel as the best strategy to the other players. In other words, since CCS and CS deviate Nash equilibria, each player is difficult to select one channel as the best strategy to the other players in our stochastic channel selection games. We observe that CRCS achieves the higher throughput than RCS. This is because CRCS gives a higher expected payoff than RCS when they are in Nash equilibria. Furthermore, when the number of node increases, the number of wireless nodes sharing the same channel N in Eq. (3.15) also increases. Therefore, we note that CRCS becomes more effective in Fig. 3.5 comparing to Fig. 3.4. We also note that CCS can even achieve a little higher performance than RCS when the distance of nodes increases in Fig. 3.5(b). This indicates that to avoid selecting the active channel as a new channel is more efficient when nodes are not in the same transmission range.

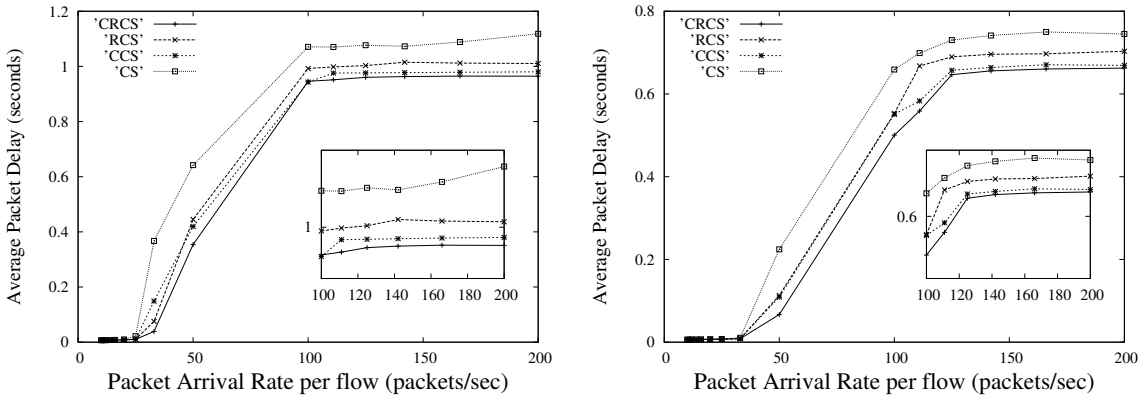
We also show average packet delay of different channel selection strategies in 16-node and 36-node topologies with three channels in Figs. 3.6 and 3.7, respectively. It should be noted that CRCS and RCS have lower average packet delay than CCS and CS, respectively. This is because CRCS and RCS can converge to Nash equilibrium. We also note that CRCS gives the lower delay than RCS. The reason also is that CRCS can provide the higher expected payoff than RCS.



(a) The distance of two nodes is $140m$.

(b) The distance of two nodes is $200m$.

Figure 3.6: Average packet delay vs. packet arrival rate in 16-node and three-channel topology



(a) The distance of two nodes is $140m$.

(b) The distance of two nodes is $200m$.

Figure 3.7: Average packet delay vs. packet arrival rate in 36-node and three-channel topology

3.7 Conclusions

In this chapter, we have introduced two kinds of stochastic channel selection strategies for multi-channel MAC protocol using single interface. One is selecting a new channel from all available channels with some probability distribution. The other is selecting a new channel from all available channels except the active channel with some probability distribution. By using game theory, we have shown the existence of a stable point, i.e., selecting a new channel with equal probability for each strategy. It should be emphasized that the higher

expected payoff can be obtained if a new channel is selected from available channels other than the active channel. Then, we have designed the channel selection algorithm that helps wireless nodes operate the stable point. We have validated the game theoretic analysis by computer simulation.

3.8 Conditionally Randomized Channel Selection Plus (CRCS+)

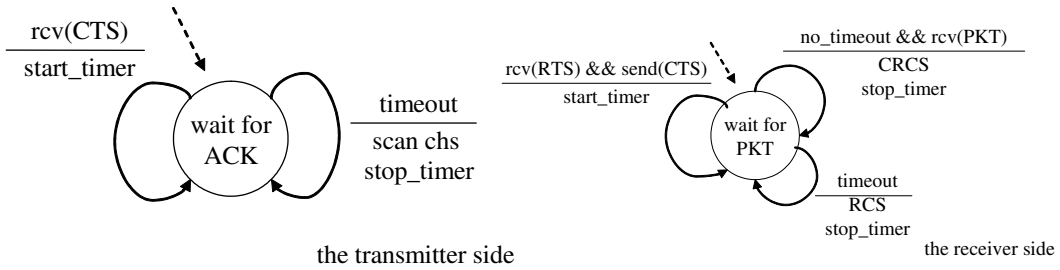
We recognize that CRCS has the rate of packet loss lower than that of RCS. It leads that CRCS can provide higher performance than RCS. Therefore, designing a channel selection algorithm that can reduce the number of packet collisions is a straightforward way to improve the performance of ad hoc networks. In this section, we extend our work to introduce Conditionally Randomized Channel Selection Plus (CRCS+) at first. Then, we also use game theory to show CRCS+ can give higher expect payoff than CRCS. Furthermore, we through computer simulation show that CRCS+ can provide the higher performance than CRCS.

To investigate CRCS+, we use the same game formulation as we described in this chapter except that a new stochastic channel selection strategy C_3 is used in strategic phase. In C_3, the player selects one channel from all available channels except the active channel with some probability if the data packet can be received successfully and selects one channel from all available channels with some probability if the data packet cannot be received.

By using Proposition 1, we can show that the following channel selection strategy can lead to a totally mixed-strategy Nash equilibrium. After independently and randomly selecting one channel from all of available channels with equal probability for the first time transmission, each player selects a new channel from available channels except the active channel *with equal probability* if the data packet can be received successfully and selects a new channel from all of available channels *with equal probability* if the data packet can not be received. We call this channel selection strategy as Conditionally Randomized Channel Selection Plus (CRCS+).

We can clearly observe that CRCS+ can give the lower collision probability than CRCS. This is because that the collision probability of CRCS+ is equal to $1/m$ while that of CRCS is equal to $1/(m-1)$ after the data packet cannot be received, and they have the same collision probability after the data packet can be received successfully. Under this result, we can expect that CRCS+ can provide the higher expected payoff than CRCS. Furthermore, we can calculate the payoff of each player using CRCS+ for 2-players and m-channels stochastic chance-selection game using CRCS+. That is equal to

$$q_n = 4\left(\frac{m-1}{m}\right) - 2 - 2c + \frac{2}{m^2(m-1)}. \tag{3.16}$$



(a) CRCS+: the transmitter side.

(b) CRCS+: the receiver side.

Figure 3.8: Finite State Machine (FSM) definitions for the transmitter and the receiver of our multi-channel protocol using CRCS+

See Appendix D for the proof of Eq. (3.16). It can be clearly recognized that CRCS+ can provide the higher expected payoff than CRCS. Moreover, the expected payoff for n-players and m-channels stochastic chance selection game is left as the future work.

3.8.1 Multi-channel MAC protocol using CRCS+

One issue for designing multi-channel MAC protocol using CRCS+ is how to decide that the data packet cannot be received successfully by the receiver. For this purpose we set two different timeouts, NAV_1 and NAV_2, on the transmitter and the receiver. We assume

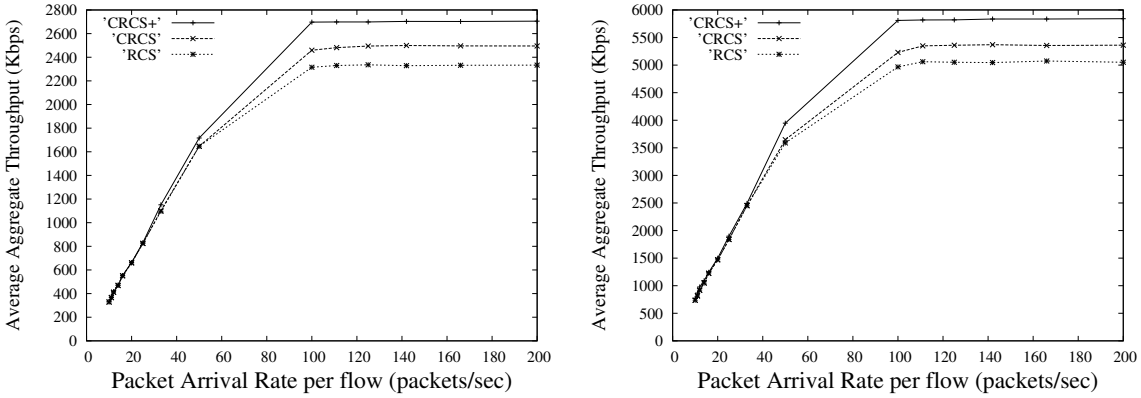
$$NAV_1 = 2sifs + txt_{PKT} + txt_{ACK} + 2txt, \tag{3.17}$$

and

$$NAV_2 = sifs + txt_{PKT} + txt, \tag{3.18}$$

where $sifs$ is the time for Short Inter Frame Space, txt_{RKT} is the transmitting time for the data packet, txt_{ACK} is the transmitting time for ACK and txt is the maximal time for the propagation delay.

As shown in Fig. 3.8(a), the transmitter side of our multi-channel MAC protocol using CRCS+ receives CTS via the rcv(CTS) event, after that it uses the $startTimer$ action for NAV_1. When the transmitter does not receive ACK in *timeout* time that equals to NAV_1, the transmitter restarts to scan all the channels for the next transmission. On the other hand, as shown in Fig. 3.8(b), the receiver side of our multi-channel MAC protocol using CRCS+ receives RTS via rcv(RTS) event and transmits CTS via send(CTS) event, after that it uses $startTimer$ action for NAV_2. When the receiver receives the data packet via rcv(RKT) in *timeout* time that equals to NAV_2, the receiver decides to use CRCS to



(a) 16-node topology with three channels.

(b) 36-node topology with three channels.

Figure 3.9: Average packet delay vs. packet arrival rate

selection a new channel. If the receiver does not receive the data packet in *timeout* time, the receiver assumes that data packet cannot be received successfully and it decides to use RCS to select a new channel.

3.8.2 Simulation results

Figures 3.9(a) and 3.9(b) show the average aggregate throughput of CRCS+, CRCS, and RCS in 16-node and 36-node topologies as increasing input packet arrival rate, when three channels are used. As shown in these figures, when we increase the arrival rate, we can observe that CRCS+ achieves the higher throughput than CRCS and RCS. This is because CRCS+ gives a higher expected payoff than CRCS and RCS when they are in Nash equilibria. This indicates that CRCS+ is a more effective channel selection strategy than CRCS and RCS.

Chapter 4

TCP over Multi-channel MAC Protocol

It is well known that the legacy Transmission Control Protocol (TCP) does not adapt to IEEE 802.11 Medium Access Control (MAC) protocol in ad hoc networks. In this chapter, we introduce the multi-channel MAC protocol who employs CRCS as its channel selection algorithm, adds a simple tuning of TCP congestion window dynamics and a cross-layer scheme. We show through computer simulation that this approach can improves TCP throughput and fairness in multi-channel ad hoc networks.

Note: The materials in this chapter have been appeared in [47].

4.1 Background

The multi-channel MAC protocols is designed for TCP in [28]. This protocol belongs to asynchronous scheme and likes DCA. However, authors only focus on mitigating wireless link-layer contention through bidirectional RTS/CTS channel reservation to reduce TCP DATA-ACK collisions. Therefore, we consider that their protocol cannot improve TCP performance for ad hoc networks so much. In fact, their simulation results in [28] show TCP throughput maximally gains 180% comparing to IEEE 802.11 MAC protocol in static multi-hop networks with four available channels (one channel is control channel and three channels are data channels)

In this chapter, our point of view is from designing an adapted TCP mechanism that employs CRCS as its channel selection algorithm to improve the performance of ad hoc networks. OUr simulation results show that TCP throughput maximally gains more than two times comparing to IEEE 802.11 MAC protocol in static multi-hop networks with three available channels. Furthermore, we extend our simulation results into more complex topologies, i.e., grid topology and mobile topology.

4.2 The window adaptation mechanism of TCP

It is well known that the rate at which data is transmitted increases linearly in time until a packet loss is detected in TCP flow control mechanism. However, most TCP implementations use a coarse timer for the detection of packet losses. For example, Fast Retransmit algorithm in TCP detects packet losses via duplicate ACKnowledgements (ACKs). This coarse timer incurs some idle times after the packet loss during which the congestion window is not increasing. We call the packet loss followed an idle time the TimeOut loss as shown in Fig. 4.1. We believe that the phenomenon still happens in TCP over ad hoc networks. Therefore, TCP performance can be improved by reducing the packet loss and the TimeOut loss events.

4.2.1 TCP congestion window dynamics

We introduce a mechanism to reduce the packet loss. In [32], it is pointed out that wireless links cannot hold more than one packet 'back-to-back' like wired links. It also is shown by simulation in [48] that the TCP performance degrades if Congestion Window Limit (CWL), which is the upper bound of TCP's congestion window size, is greater than one or two packets. These observations lead us to modify the TCP congestion window dynamics presented below.

First, we design our protocol to allow TCP senders to inject small number of packets

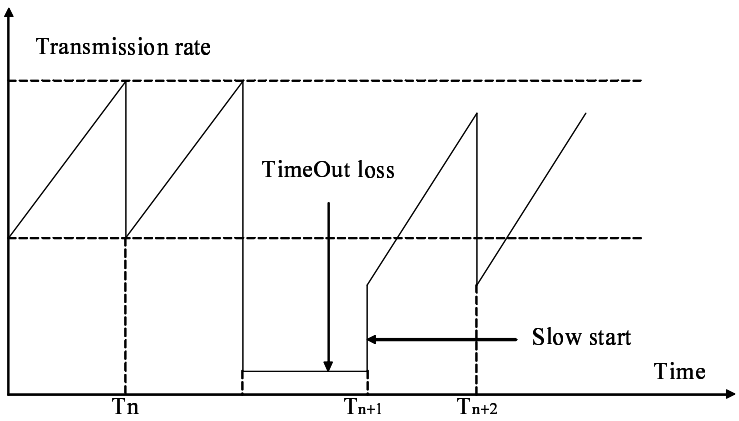


Figure 4.1: A model for TCP with TimeOut event.

into wireless link at one time. We abandon the slow-start mechanism in legacy TCP for this purpose. This is because the slow-start exponentially increases the congestion window size and easily overshoots the 'optimal' value of CWL beyond which the performance degrades. Instead, the congestion window size is governed by the congestion avoidance phase at the beginning. Second, we also change the growth rate of the congestion window in the congestion avoidance phase of legacy TCP. Starting from the minimum size, our algorithm increases the congestion window gradually, much more slowly increasing than one Maximum Segment Size (MSS) every Round Trip Time (RTT). More specifically, every time a TCP-ACK is received, the TCP sender in our algorithm updates the size of the congestion window *cwnd* by

$$cwnd \leftarrow cwnd + \alpha/cwnd, \tag{4.1}$$

where α is a small positive constant much less than one. We will show that this mechanism can reduce the packet loss rate in our computer simulation.

4.2.2 Cross-layer scheme

We also introduce a scheme to reduce the TimeOut loss event in TCP layer. To this end, our protocol adopts a cross-layer scheme that enables MAC layer to send a retransmission message to TCP layer when the transmitter detected the packet loss at MAC layer. Note that the scheme of sending the retransmission message is invoked on the transmitter node, not forwarding or receiving nodes. If TCP layer of the transmitter receives this message, the congestion window size is set to one and then sends the data packet as soon as possible. We also prove that our cross-layer scheme can reduce the TimeOut loss event via our computer simulation.

4.3 Performance evaluation

We have conducted extensive simulations using the ns-2 network simulator [35] with CMU wireless extensions to evaluate the TCP performance of our multi-channel MAC protocol. For the simplification, we call our multi-channel MAC protocol based on the method congestion window limit Muti-Channel MAC (cwl-MCMAC) protocol. The performance of cwl-MCMAC is compared with Dynamic Channel Allocation (DCA) [4] that potentially has a bottleneck due to only one dedicated control channel. We also evaluate the performance of MCMAC, which uses only the channel selection algorithmof cwl-MCMAC, to confirm the effects of our TCP window dynamics and the cross-layer scheme. We use the common parameters of cwl-MCMAC and MCMAC in each simulation experiment. For the channel selection strategy, we assume that all available channels have the same threshold th_c equal to 100 packets. We set the growth rate $\alpha = 0.01$ in our modified TCP congestion window dynamics.

Each simulation lasts for 200 seconds. All flows start at the same time after one second. We assume the following common parameters in each experiment; all the radio parameters are ns-2 defaults; the data rate of the wireless channel is 2 Mbps; the radio propagation mode is the two-ray ground model with transmission range 250m; carrier sensing and interference ranges are both 550m; all of nodes have queueing buffer of size 50 packets. We set up a single TCP-Reno [49] connection between a chosen pair of sender and receiver nodes. All of topologies represent the packet forwarding path generated by a minimum-hop routing protocol such as DSR [50].

4.3.1 Chain topology I

Our first TCP performance evaluation assumes the chain topology with nine hops aligned in line as illustrated in Fig. 4.2. The distance between neighboring nodes is 200m. We consider two TCP flows transmitted in three different patterns from different nodes. They are:

1. node 4 $\Rightarrow$ node 0, node 5 $\Rightarrow$ node 9,
2. node 0 $\Rightarrow$ node 4, node 9 $\Rightarrow$ node 5,
3. node 0 $\Rightarrow$ node 9, node 9 $\Rightarrow$ node 0.

Three channels are available for all of multi-channel protocols.

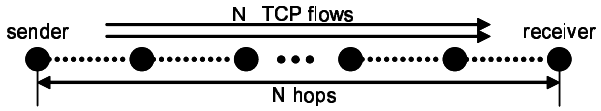


Figure 4.2: Chain topology.

Table 4.1: The fairness index of line topology

Protocol	4⇒0,5⇒9	0⇒4,9⇒5	0⇒9,9⇒0
IEEE 802.11	0.9683	0.9637	0.6001
DCA	0.9804	0.9916	0.8774
MCMAC	0.9809	0.9678	0.6271
cwl-MCMAC	0.9999	0.9995	0.6516

4.3.1.1 Fairness

To evaluate the fairness issue, we adopt the Jain's fairness index ξ [36], which is defined by

$$\xi = \frac{(\sum_{i=1}^{N} T_i)^2}{N \sum_{i=1}^{N} T_i^2}, \tag{4.2}$$

where T_i is the throughput of TCP connection i, and N is the total number of connections. The fairness index takes the value between 1 and $1/N$. When ξ is equal to 1, the fairness is the best. Oppositely, the fairness is the worst if ξ is equal to $1/N$.

The fairness indexes of TCP flows with our protocols, DCA, and IEEE 802.11 are shown in Table 4.1. We can find that cwl-MCMAC, MCMAC, and DCA achieve the higher fairness index than IEEE 802.11. This is because using multiple channels can reduce the contention of wireless channel and mitigate the specific TCP flow starvation. We also find that cwl-MCMAC and MCMAC are not always better than DCA. The main reason is that our protocols change the channel to transmit packets when the receiver considers that the quiescent channel is saturated, while DCA changes the channel to transmit each packet as the channel negotiation is done by packet-by-packet. However, it should be noticed that the cost of packet-by-packet channel negotiation is too expensive for the whole transmission.

4.3.1.2 Throughput

As shown in Table 4.1, we see that the fairness index of DCA is better than our protocols for flows of node 0 ⇒ node 9 and node 9 ⇒ node 0. Using the same topology with that in

Table 4.2: Throughput (Kbps)

IEEE 802.11	DCA	MCMAC	cwl-MCMAC
38.847	72.064	73.979	83.9408

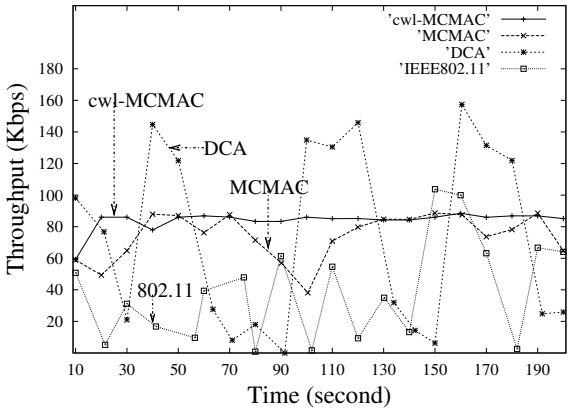


Figure 4.3: Instantaneous throughput of different protocols in the line topology.

Fig. 4.2, we examine whether DCA can provide the higher throughout than our protocols. We show results in Table 4.2 and Fig. 4.3.

From Table 4.2, we can find that the throughput of IEEE 802.11 is the worst one. The reason is that certain nodes cannot receive TCP data packets correctly, as IEEE 802.11 can cause wireless link-layer contention with neighboring nodes. According to the analysis in [27], the maximum spatial reuse of a chain topology of nodes is only 1/4 of the number of hops. Although this kind of contention still occurs in multi-channel networks, DCA allows nodes to select a suitable channel from available channels to avoid strongly impacting on a unique channel for TCP data packets. Therefore, DCA improves aggregate throughput by 185.51%, comparing to IEEE 802.11.

Since all nodes are static, there are no routing failures due to mobility, and hence the instantaneous throughput should be stable. As shown in Fig. 4.3, however, the instantaneous throughput of DCA seriously fluctuates and frequently drops to zero. The primary reason is that a node is unable to learn the situation of its neighbors' channels in DCA if it listens to the different channel. This makes it difficult to use virtual carrier sensing (e.g., RTS and CTS) to avoid wireless link-layer contention due to the hidden terminal problem. Therefore, the hidden terminal problem cannot be eliminated in DCA. Because the hidden terminal problem can induce wireless link-layer contention where senders cannot distinguish whether broadcast packets are successful or not, it can cause the throughput of TCP flow fluctuates

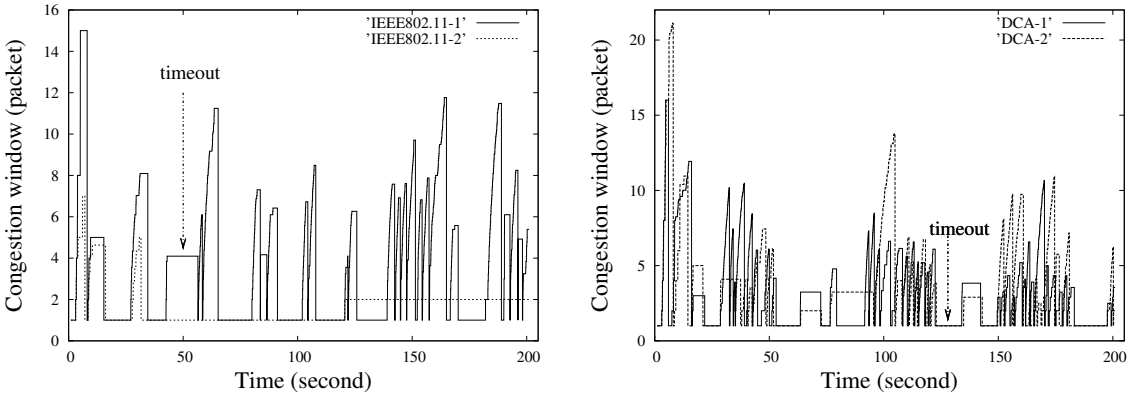
severely and even frequently drop to zero. This problem is called TCP instability [30].

In contrast, MCMAC can mitigate the hidden terminal problem thanks to using busy tone. This is because the hidden terminal problem can be alleviated in MCMAC as any other potential transmitters defer to send packets by sensing on the busy tone. In addition, because MCMAC uses CRCS as its channel selection algorithm to increase and balance the utilization of channels, the fluctuation becomes less severe than IEEE 802.11 and DCA. However, MCMAC still gets the penalty of the legacy TCP's Additive-Increase and Multiplicative-Decrease (AIMD) congestion window algorithm. Comparing to MCMAC, cwl-MCMAC can reduce the penalty as it keeps its congestion window size small. Therefore, the TCP instability is overcome by cwl-MCMAC. In fact, Fig. 4.3 shows that instantaneous throughput of cwl-MCMAC is kept stable. At the same time, cwl-MCMAC can achieve 216.08% performance that gains over IEEE 802.11 in terms of aggregate throughput.

To ascertain the stability of the throughput in cwl-MCMAC, we show the TCP congestion window behaviors over IEEE 802.11, DCA, MCMAC, and cwl-MCMAC for the same topology in Figs. 4.4 and 4.5. The crucial observation in Figs. 4.4(a) and 4.4(b) are that the congestion window rapidly grows or decreases, and is kept constant during certain amount of time, possibly due to the TimeOut loss event, for IEEE 802.11 and DCA. Comparing to them, the congestion window size in MCMAC and cwl-MCMAC are small as shown in Figs. 4.5(a) and 4.5(b). This is because our cross-layer scheme enables TCP layer to restart the congestion window dynamics by receiving a retransmission message that is immediately sent when the transmitter detected the packet loss at MAC layer. However, since MCMAC still uses the slow-start mechanism in legacy TCP, the congestion window grows exponentially when TCP congestion window dynamics is restarted. In other words, the congestion window size in MCMAC increases quickly from one packet to more than two packets as shown in the inset of Fig. 4.5(a). Injecting more than or equal to two packets in wireless link easily induces the packet loss as wireless link cannot hold more than one packet 'back-to-back' like wired links. Therefore, the congestion window size in MCMAC is unstable, as shown in Fig. 4.5(a).

On the other hand, in cwl-MCMAC, we abandon the exponential growth of the congestion window by slow-start mechanism in legacy TCP and we increase the congestion window so gradually that at most two packets are always injected into wireless link in our scenario, as shown in the inset of Fig. 4.5(b). In addition, via collaborating MAC and TCP layers, cwl-MCMAC immediately knows the packet loss at MAC layer and promptly re-sends the packet. Therefore, the congestion window in cwl-MCMAC is stable as shown in Fig. 4.5(b).

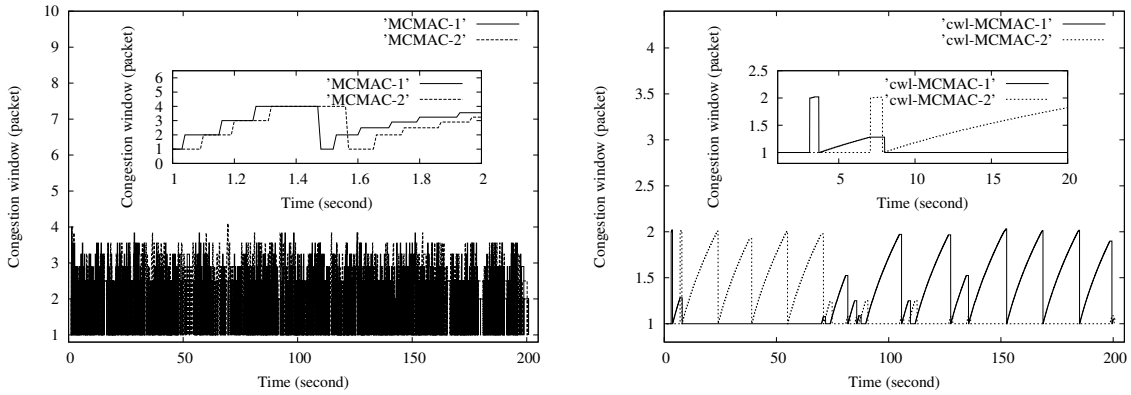
Moreover, we should notice that all of nodes have a queueing buffer whose size is 50 packets. If the packet loss event is caused by buffer overflow before wireless link-layer contention, then the congestion window size could be increased to 50 packets or more. From Fig. 4.4, we find that the congestion window size reaches at most about 20 packets.



(a) IEEE 802.11

(b) DCA

Figure 4.4: TCP congestion window behaviors in IEEE 802.11 and DCA.



(a) MCMAC

(b) cwl-MCMAC

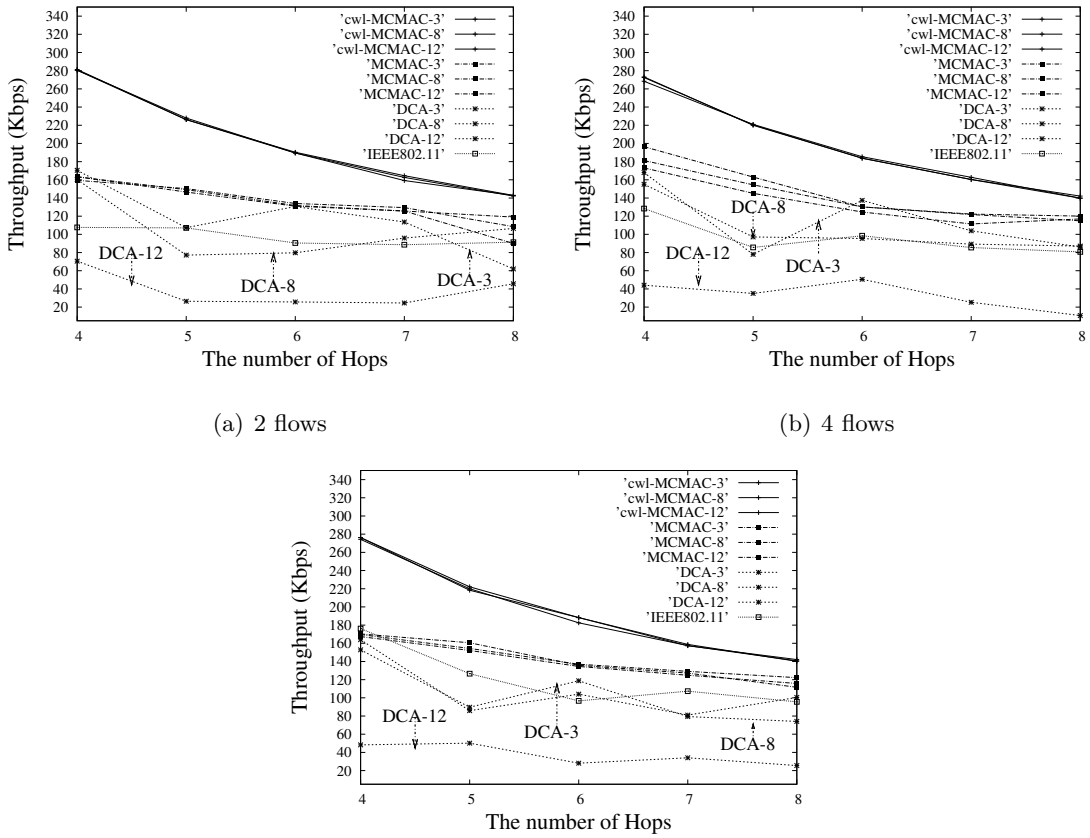
Figure 4.5: TCP congestion window behaviors in MCMAC and cwl-MCMAC.

This phenomenon reconfirms us the result of [27], which indicates that the network overload is typically first signified by packet loss due to wireless link-layer contention, rather than packet loss at buffer overflow.

4.3.2 Chain topology II

In our second chain topology, we examine whether our channel selection algorithm can improve the efficiency of multiple channels, and also evaluate the window adaptation mechanism of TCP for cwl-MCMAC still works well.

In this subsection, TCP performance evaluation assumes the static chain topology with



(a) 2 flows

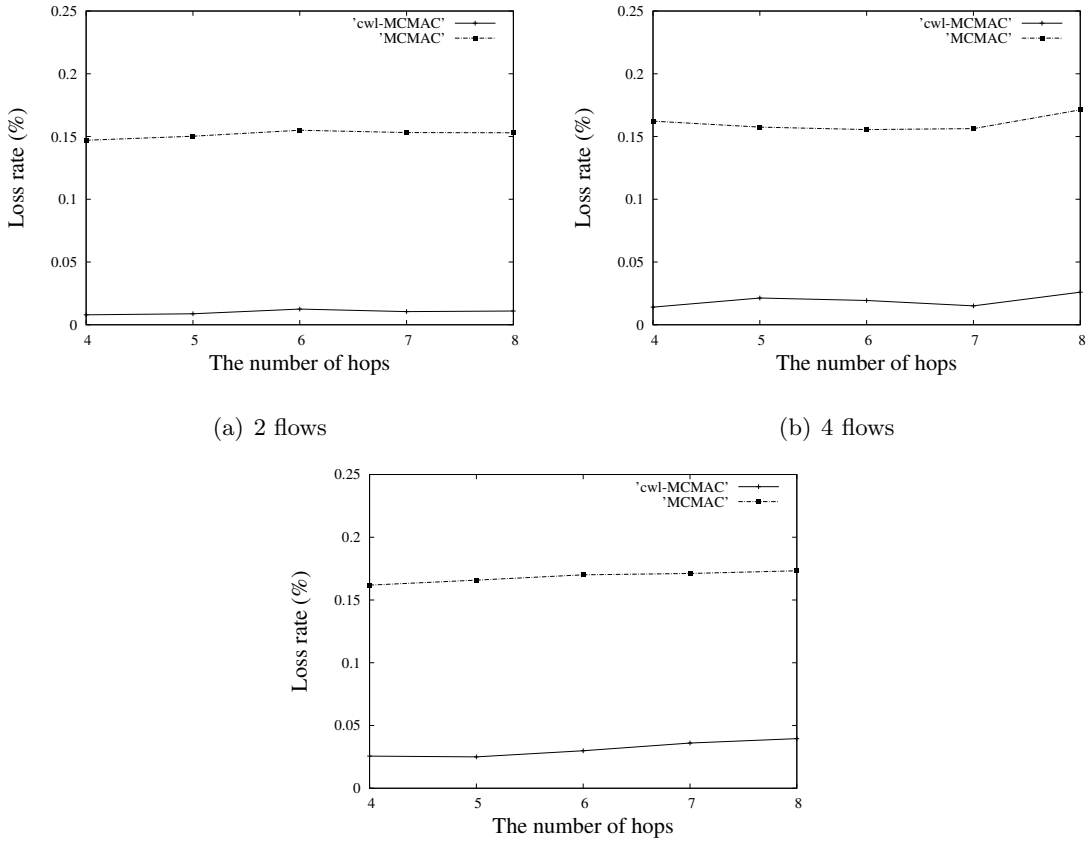
(b) 4 flows

(c) 8 flows

Figure 4.6: TCP throughput versus the number of hops with 2, 4, and 8 TCP flows in the chain topology.

different hops and flows as illustrated in Fig. 4.2. The distance between neighboring nodes is also 200m. The number of hops is from four hops to eight hops and the number of flows is two, four, or eight. We consider that TCP flows are originated by first node and are forwarded to the last node. All flows also start at the same time. Three, eight, and twelve channels are available for all of multi-channel protocols. Note that the symbol '-3' means three channels are available for the multi-channel MAC protocol. For example, cwl-MCMAC-3 means that three channels are available for cwl-MCMAC.

To illustrate our channel selection algorithm can efficiently utilize the multiple channels, we compare the TCP throughput of IEEE 802.11, DCA, MCMAC, and cwl-MCMAC in Fig. 4.6. From these figures, we can observe that the performance of DCA becomes worse as increasing available channels. In DCA, since the hidden terminal problem still occurs and the channel for transmitting the data packet is changed by packet-by-packet, too many



(a) 2 flows

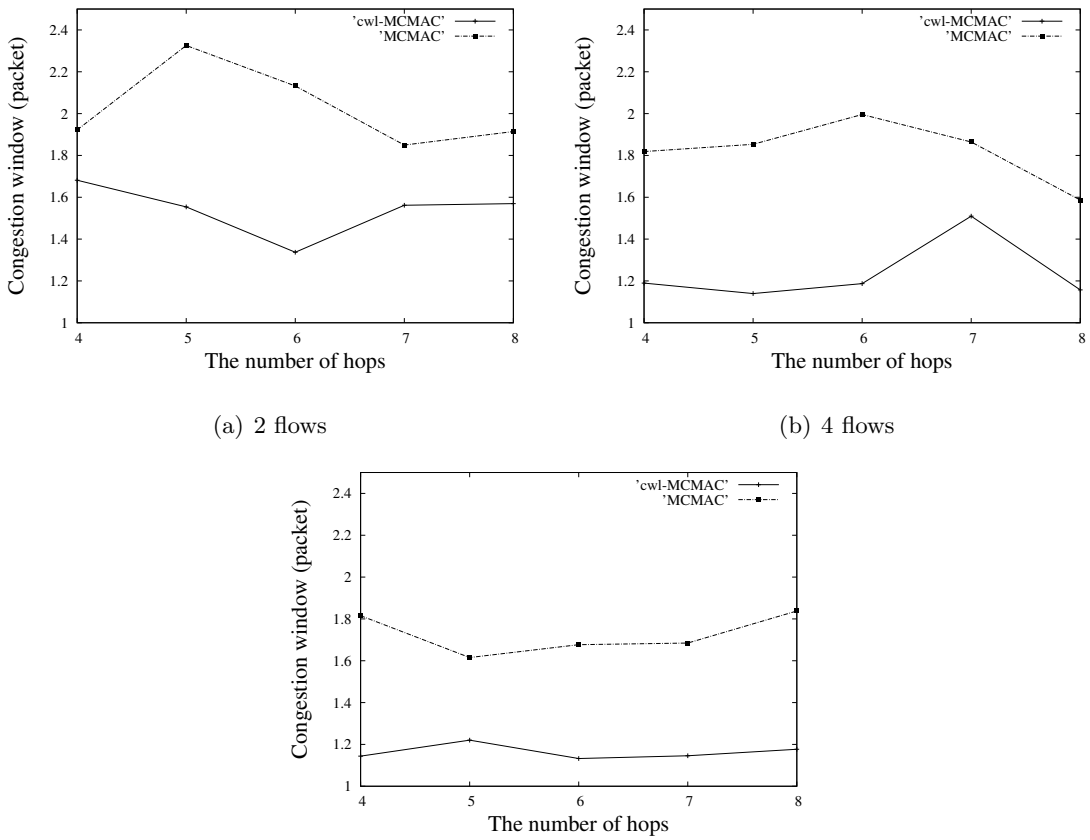
(b) 4 flows

(c) 8 flows

Figure 4.7: The average loss rate versus the number of hops with 2, 4, and 8 TCP flows in the chain topology. There are three available multiple channels for all multi-channel MAC protocols.

control messages are injected into the one dedicated control channel. Those make that the control channel saturated easily. Because the same things still take place in Figs. 4.6(b) and 4.6(c), the performance deteriorates compared to the case of Fig. 4.6(a), as the saturation of control channel happens more easily if the number of flows, hops, and channels increase.

Meanwhile, our protocols are much better than DCA. This is because busy tones as control messages spread over all channels based on our channel selection algorithm, not concentrate on one control channel, as we assume that a different busy tone is assigned on each channel. Therefore, cwl-MCMAC and MCMAC can alleviate the problem of control channel bottleneck to increase the efficiency of multiple channels by using CRCS as their channel selection algorithms. Between these two protocols, cwl-MCMAC can provide the



(a) 2 flows

(b) 4 flows

(c) 8 flows

Figure 4.8: The average congestion window size versus the number of hops with 2, 4, and 8 TCP flows in the chain topology. There are three available channels for all multi-channel MAC protocols.

higher performance than MCMAC as cwl-MCMAC reduces the packet loss event by maintaining the small congestion window size. To prove that, we show the average packet loss rate and congestion window size of MCMAC and cwl-MCMAC in Figs. 4.7 and 4.8. We can find that cwl-MCMAC has much less loss rate than MCMAC as shown in Fig. 4.7. This is because cwl-MCMAC can always keep one packet on wireless link to reduce the packet loss event even when the number of hops and flows increases, as shown in Fig. 4.8.

In Fig. 4.6, we find out that although cwl-MCMAC does not deteriorate TCP throughput when the number of available channels increases in chain topology, cwl-MCMAC cannot significantly improve it. Recall that the interference range is 550m and the one-hop distance is 200m in our simulation. For avoiding interference, the first three hops need to select different channels. Since the fifth node is out of the interference range of the first node,

Table 4.3: The flow patterns in 7x7 grid topology.

The number of flows	flow pattern
2 flows	3 ⇒ 45, 21 ⇒ 27
4 flows	1 ⇒ 43,5 ⇒ 47, 7 ⇒ 13,35 ⇒ 41
6 flows	1 ⇒ 43,3 ⇒ 45, 5 ⇒ 47, 7 ⇒ 13,21 ⇒ 27,35 ⇒ 41
8 flows	0 ⇒ 42,2 ⇒ 44,4 ⇒ 46,6 ⇒ 48, 0 ⇒ 6,14 ⇒ 20,28 ⇒ 34,42 ⇒ 48
10 flows	randomly generate 10 flows

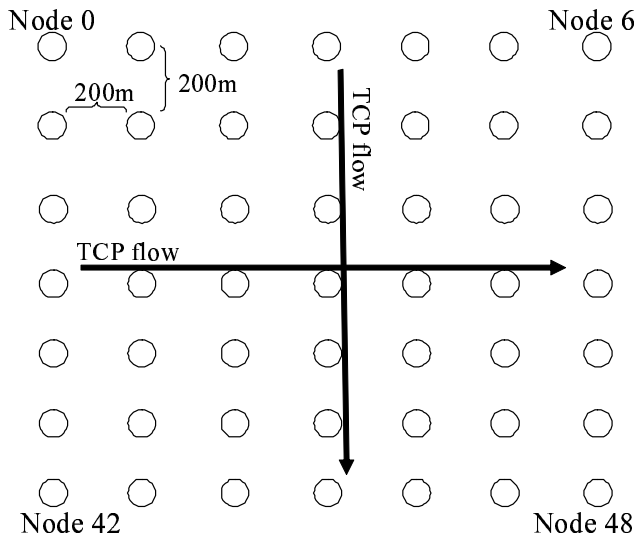
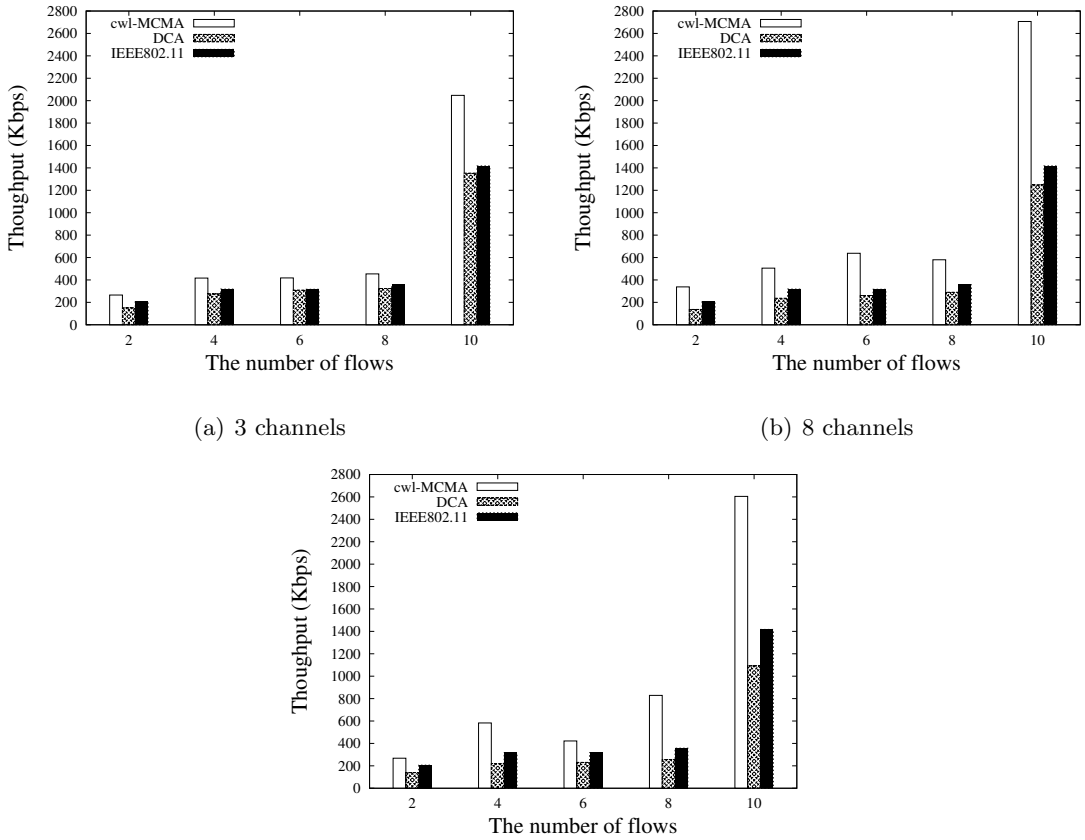


Figure 4.9: 7x7 grid topology.

the channel used in the first hop can be selected in the fourth hop without the interference, and hence three channels are enough for this chain topology. Because three channels are sufficient to maximize the concurrency in the chain topology, multiple channels cannot improve TCP throughput.

4.3.3 Grid topology

We extend our study to scenarios of multiple TCP flows and channels in grid topology. Figure 4.9 shows a 7x7 grid topology. We run 2, 4, 6, 8 and 10 flows, respectively. The flow patterns are shown in Table 4.3. We keep the simulation parameters that are the same as



(a) 3 channels

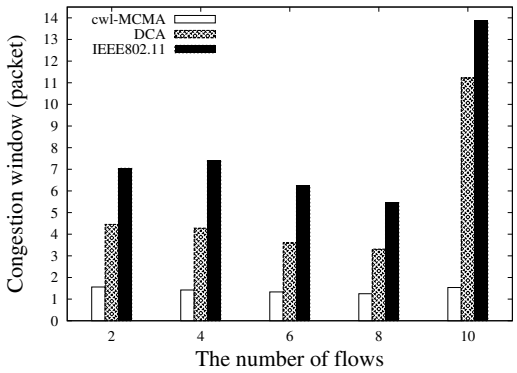
(b) 8 channels

(c) 12 channels

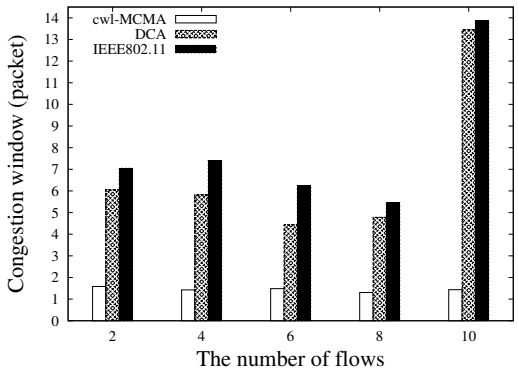
Figure 4.10: TCP throughput versus the number of flows in 7x7 grid topology.

the chain topology.

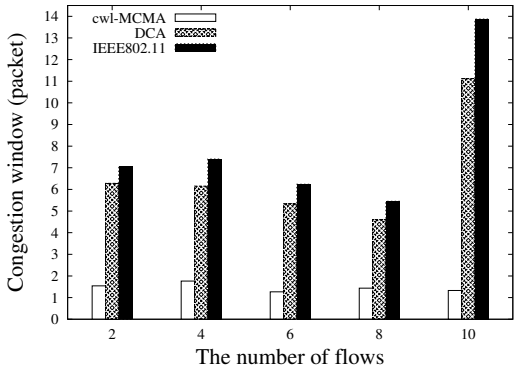
The TCP throughput versus the number of available flows in various multiple channels in 7x7 grid topology are shown in Fig. 4.10. We find that DCA clearly gives worse performance than IEEE 802.11, especially when the number of channels increases. The main reason for DCA not to improve its performance so much is that DCA cannot increase its the efficiency of multiple channels, due to the bottleneck in control channel. From Fig. 4.10, we reconfirm that increasing the efficiency of multiple channels through channel selection algorithm is an important factor to improve TCP performance in ad hoc networks. In contrast, cwl-MCMAC can gain the higher throughput performance than others. This is because it can increase the efficiency of multiple channels by using CRCS as its channel selection algorithm, and can mitigate the wireless link-layer contention by using busy tone. Furthermore, it can also mitigate the wireless link-layer contention by using the window adaptation mechanism of TCP. Actually, as shown in Fig. 4.11, we can find that our window adaptation mechanism of TCP cannot get any effect even that the number of multiple channels increases.



(a) 3 channels



(b) 8 channels



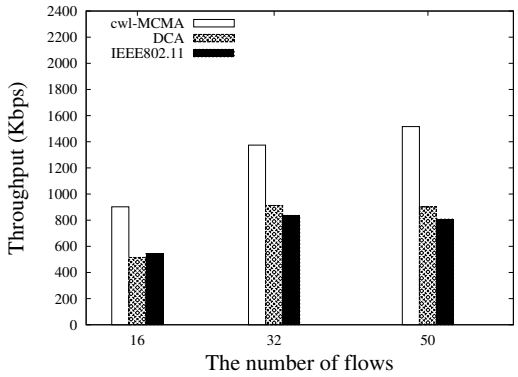
(c) 12 channels

Figure 4.11: The average size of congestion window versus the number of flows in 7x7 grid topology.

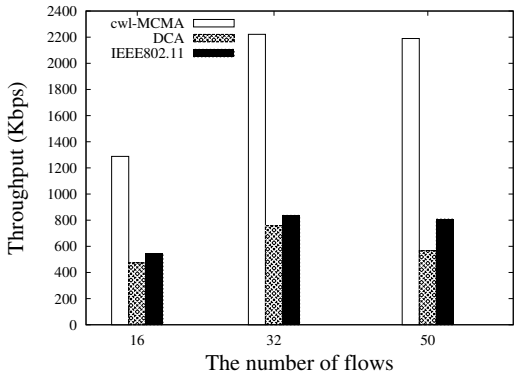
4.3.4 Mobile topology

We also extend our simulation with mobile topology. 150 mobile nodes are placed randomly in an area of 2,000m x 2,000m. We consider the cases where there are 16, 32, and 50 TCP flows. Their sources and destinations are randomly chosen. The mobile nodes move according to the random waypoint model.

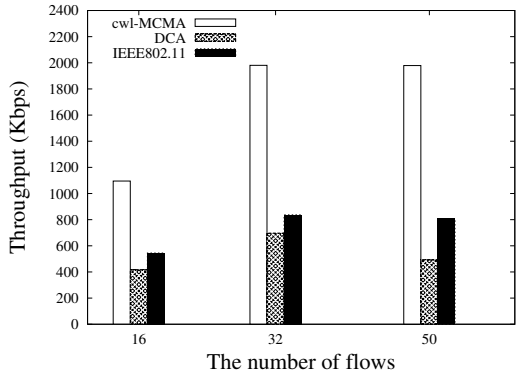
In mobile topology, cwl-MCMAC yields the similar trend of performance improvement of TCP throughput with cwl-MCMAC in 7x7 grid topology, as shown in Fig. 4.12. Since the wireless link-layer contention is more frequent and persistent in mobile ad hoc networks than static ad hoc networks, DCA using multiple channels can gain a little higher TCP throughput than IEEE 802.11, as shown in Fig. 4.12(a). However, we can also find out that the TCP performance of DCA gets worse than IEEE 802.11 when the number of channel increases. This is because DCA cannot improve its efficiency of multiple channels due to the



(a) 3 channels



(b) 8 channels



(c) 12 channels

Figure 4.12: TCP throughput versus the number of flows in mobile topology.

bottleneck of the control channel, as shown in Figs. 4.12(b) and 4.12(c). Furthermore, we can recognize that cwl-MCMAC provides the best throughput among the three protocols, since it can increase the efficiency of multiple channels by using CRCS as its channel selection algorithm.

4.4 Conclusions

We have proposed a multi-channel MAC protocol called cwl-MCMAC to enhance the TCP performance over multi-channel ad hoc networks. Our protocol is based on xRDT protocol that exploits busy tone. We employ CRCS as its channel selection algorithm to increase and balance the utilization of available channels, and to alleviate the control channel bottleneck. In our protocol, the TCP congestion window dynamics is modified to keep the window size small. We also have provided a cross-layer scheme between MAC and TCP layers to reduce

the timeout events in TCP layer. Simulation results have demonstrated that our protocol cwl-MCMAC can achieve the higher TCP throughput and the better fairness.

Chapter 5

Conclusions

This thesis focuses on the impact of multiple channels in ad hoc networks. In the Chapter 2, we design a channel selection algorithm called Conditionally Randomized Channel Selection (CRCS) base on Extended Receiver Directed Transmission (xRDT) protocol that only uses one packet interface. By computer simulation, we show that CRCS is effective to improve the performance of ad hoc networks. In the Chapter 3, we use game theory to investigate CRCS and find that CRCS is the existence of a stable point. Furthermore, we extend CRCS to design a new channel selection algorithm called Conditionally Randomized Channel Selection Plus (CRCS+) and find CRCS+ is more effective than CRCS. In the Chapter 4, we propose a multi-channel MAC protocol called congestion window limit Multi-Channel MAC (cwl-MCMAC) protocol that employs CRCS as its channel selection algorithm. By computer simulation, we also show that cwl-MCMAC can provide good TCP performance of ad hoc networks.

Appendix A

IEEE 802.11 WLANs Standard

The IEEE introduced WLANs standards with the creation of the 1997 ratification of the 802.111 standard, which has been replaced by more advanced standards. In order of ratification, the standards are 802.11a, 802.11b, and 802.11g. Table A.1 lists some key points about the currently ratified standards.

A.1 Wireless Modes of Operation

WLANs can use one of two modes:

- **Ad hoc mode:** With ad hoc mode, a wireless device wants to communicate with only one or a few other devices directly, usually for a short period of time. In these cases, the devices send WLANs frames directly to each other.
- **Infrastructure mode:** In infrastructure mode, each device communicates with a wireless access point (AP), with the AP connecting via wired Ethernet to the rest of the network infrastructure. Infrastructure mode allows the WLANs devices to communicate with servers and the Internet in an existing wired network.

A.2 Wireless Encoding and Channels

Generally, there are three classes of encoding in WLANs:

- **Frequency Hopping Spread Spectrum (FHSS):** FHSS uses all frequencies in the band, hopping to different ones. By using slightly different frequencies for consecutive transmissions, a device can hopefully avoid interference from other devices that use

Table A.1: WLAN Standards

Feature	802.11a	802.11b	802.11g
Year ratified	1999	1999	2003
Maximum speed using DSSS	-	11 Mbps	11 Mbps
Maximum speed using OFDM	54 Mbps	-	54 Mbps
Frequency band	5 GHz	2.4 GHz	2.4 GHz
Channels (non-overlapped)	23 (12)	11 (3)	11 (3)
Speeds required by standard (Mbps)	6, 12, 24	1, 2, 5.5, 11	6, 12, 24

the same unlicensed band. The original 802.11 WLAN standards used FHSS, but the current standards do not.

- **Direct Sequence Spread Spectrum (DSSS):** DSSS was designed for use in the 2.4GHz unlicensed band and is used by 802.11b. This band has 11 overlapping DSSS channels. Three of the channels (channels 1, 6, and 11) do not overlap enough to impact each other. So when designing an Extended Service Set (ESS) WLANs, APs with overlapping areas should be set to use different non-overlapping channels.
- **Orthogonal Frequency Division Multiplexing (OFDM):** Like DSSS, WLANs that use OFDM can use multiple non-overlapping channels. OFDM is used by 802.11a.

A.3 Medium Access Control (MAC)

The IEEE 802.11 MAC layer specifications coordinate the communications between stations and control the access to the channel. The Distributed Coordination Function (DCF) describes the default MAC protocol operations. DCF is based on a carrier sense, multiple access, collision avoidance (CSMA/CA) scheme. The MAC provides a virtual carrier sense mechanism. This mechanism is referred to as the Network Allocation Vector (NAV). The NAV maintains a prediction of future traffic on the medium. This prediction is based on the information that was sent in Request-To-Send (RTS) and Clear-To-Send (CTS) control frames. The NAV setting is shown in Fig A.1. Communication is established when one of the wireless nodes sends the RTS frame. The receiving station issues a CTS frame

that echoes the sender's address. As generally the antenna is omni-directional, all nodes within the transmission range of the sender are able to decode the RTS. Then they will update their NAV, as well as nodes within the transmission range of the receiver are able to decode the CTS. The exchange of RTS/CTS prior to current data frames provides a channel reservation, in order to avoid interference caused by hidden nodes.

According to the 802.11 protocol specifications, the MAC must implement the basic access method as shown in Fig. A.2. If a node wants to access the channel for transmission, it has to wait until the channel becomes idle through the CSMA/CA algorithm. If the channel is sensed idle for a period greater than a DCF Inter-Frame Space (DIFS), then the node goes into a random backoff time. The backoff duration, expressed in slot, is uniformly chosen in the interval $\{0, 1, \ldots, CW\}$, where contention window CW is between CW_{min} and CW_{max}. For example in 802.11b, $CW_{min} = 7$ and $CW_{max} = 1023$. If the medium is sensed busy at any time during the backoff, the backoff procedure is suspended. It is resumed after the medium has been idle for the duration of DIFS period. Upon the successful reception of a transmitted frame, the destination node returns an ACK frame after a Short Inter-Frame Space (SIFS). If an ACK is not received within an ACK timeout interval, the sender node assumes that the transmission is failed, and it needs to retransmit data frame by repeating the basic access procedure. In case of repeated transmission failures, the node can retransmit a short frame up to 7 times, and 5 times in case of a long frame. After that, the node will drop this data frame, and the MAC layer will notify Logical Link Control (LLC) layer about a link failure. In fact, a node that fails transmitting a frame assumes that the failure is due to collisions caused by other nodes transmissions, which are contending to access the channel. So to resolve the contention, before repeating the transmission procedure, a node duplicates its contention window. By duplicating the CW, the probability that two nodes select the same backoff time becomes smaller, then collisions are avoided.

A.4 Hidden Terminal Problem

In single-channel ad hoc networks, wireless nodes may rely on a physical carrier-sensing mechanism to determine an idle channel, such as in the IEEE 802.11 DCF function. This sensing mechanism does not solve completely the *hidden terminal* problem. Before explaining this problem, we need to clarify the "transmission range" term. The transmission range is the range, with respect to the transmitting station, within which a transmitted packet can be successfully received.

A typical hidden terminal situation is depicted in Fig. A.3. Nodes A and C have a frame to transmit to node B. Node A cannot detect C's transmission because it is outside

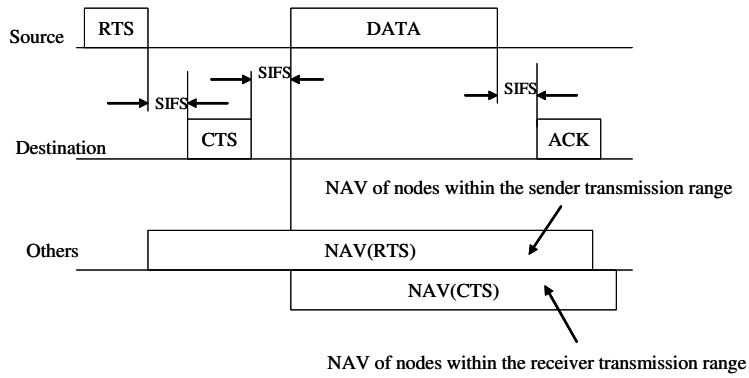


Figure A.1: RTS/CTS/DATA/ACK and NAV setting

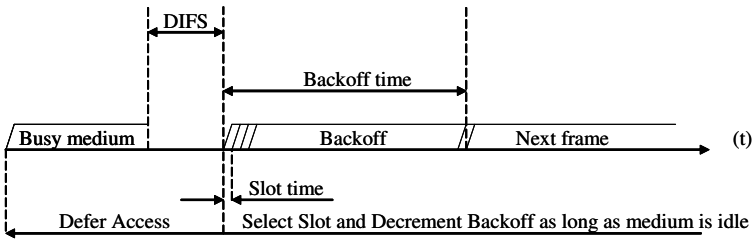


Figure A.2: Basic access method

the transmission range of C. Node C (resp. A) is therefore "hidden" to node A (resp. C). Since A and C transmission areas are not disjoint, there will be packet collisions at B. These collisions make the transmission from A and C toward B problematic. To alleviate the hidden terminal problem, virtual carrier sensing has been introduced [1]. It is based on a two-way handshaking that precedes data transmission. Specifically, the source node transmits a short control frame, called RTS, to the destination node. Upon receiving the RTS frame, the destination station replies by a CTS frame, indicating that it is ready to receive the data frame. Both RTS and CTS frames contain the total duration of the data transmission. All stations receiving either RTS or CTS will keep silent during the data transmission period.

However, as pointed out in [51], the hidden terminal problem may persist in IEEE 802.11 ad hoc networks even with the use of the RTS/CTS handshake. This is due to the fact that the power needed for interrupting a packet reception is much lower than that of delivering a packet successfully. In other words, the node's transmission range is smaller than the sensing node range.

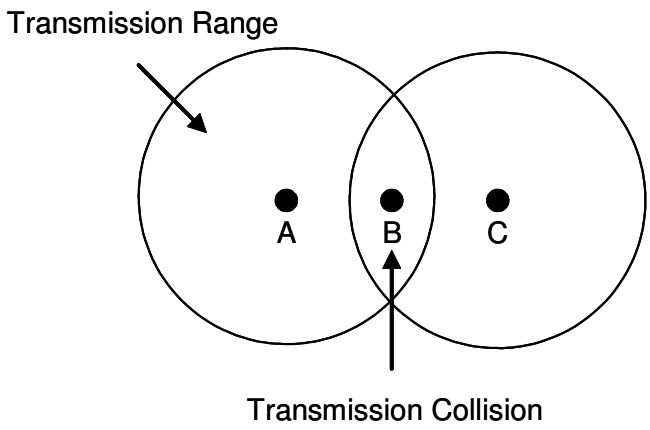


Figure A.3: Hidden terminal problem: Packets sent to B by A and C will collide at B.

Appendix B

Alleviate the hidden terminal problem for multi-channel MAC protocol by using busy tone

In this appendix, we call the multi-channel MAC protocol based on xRDT borrowing busy tone as Busy-Tone based Multiple channels MAC (BTMMAC). The main contribution of this Appendix is to show that busy tone can alleviate the hidden terminal problem in multi-channel ad hoc networks by using a Markovian modeling approach. The materials in this appendix have been appeared in [52].

B.1 Performance Analysis

The Markov model, used in [53] to study Carrier Sense Multiple Access (CSMA) and Busy Tone Multiple Access (BTMA), is adopted to evaluate the probability of the successful packet transmission of IEEE 802.11, DCA, and BTMMAC.

B.1.1 Traffic model and notation

The following is the list of assumptions for our work that uses the same network model with [53] and [54].

1. All nodes are deployed in two-dimensional Poisson distribution with density λ. Therefore, the probability that i nodes appear in an area of size A is given by

$$p(i, A) = \frac{(\lambda A)^i e^{-\lambda A}}{i!}. \tag{B.1}$$

2. We assume that the channel activity is illustrated in Fig. B.1. Time is divided into slots of duration a that is equal to the one way propagation delay. The length of data

packet is the same for all the nodes. The constant data packet transmission time is chosen as the unit of time.

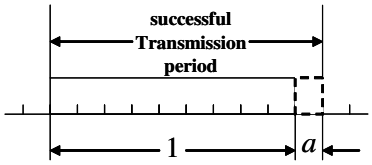


Figure B.1: Illustration of the channel activity.

3. We denote the transmission time of RTS, CTS, DATA, ACK, and Busy Tone by T_{rts}, T_{cts}, T_{data}, T_{ack}, and T_{bt}, respectively. We assume that T_{rts}, T_{cts}, T_{data}, T_{ack}, and T_{bt} are expressed as the integers multiplied by a.

4. All nodes have the same transmission power with radius R. If we denote the hearing region of node A by $N(A)$, then $N(A) = \pi R^2$. We assume $R = 1$ for the sake of simplicity.

5. Suppose the situation where node A transmits one packet to node B, while node C is out of hearing region of node A as illustrated in Fig. B.2. Since node C is a hidden node, the interference at node B may happen if node C transmits at any time during the period that starts one packet transmission time before node A begins its transmission and ends after packet transmitted by node A is received by node B. The period is called *vulnerable period*. We assume the vulnerable period as illustrated in Fig. B.3, i.e., we consider that the longest vulnerable period as the transmission time of data packet is biggest among other packets.

6. According to [53], for each slot, we introduce p as the probability that a silent node is ready to transmit a packet and assume that p is a small probability. In [54], authors have calculated the optimal value p that is equal to 0.113. We assume that nodes transmit only at the start of an even numbered slot in BTMMAC. The reason for this is that our protocol listens to busy tone for sensing the channel prior to its transmission. Therefore, the probability p_t that a node transmits a packet is given by $p_t = 2p$, where $0 < p_t < 1$. Note that p_t is approximate value for BTMMAC.

7. We also assume that there are M channels available for every node in the network with

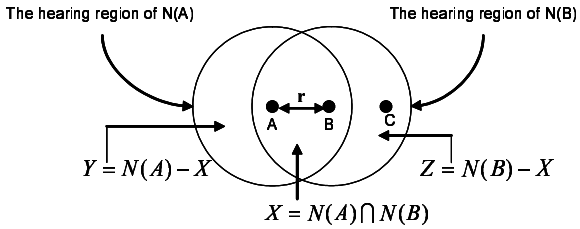


Figure B.2: Illustration of the collision in ad hoc networks

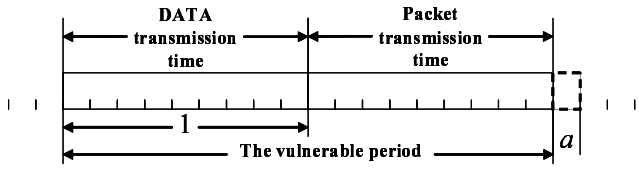


Figure B.3: The vulnerable period in ad hoc networks

equal probability. Namely, a node chooses one of the M channels with probability $1/M$, if it is ready.

B.1.2 The Markov model

Let us consider a discrete-time Markov chain on state space $\{S,\ I,\ C\}$, where S, I, and C are successful, idle, and collision states of a node, respectively. Let $P(S)$, $P(I)$, and $P(C)$ denote the steady-state probability of the successful state S, the idle state I, and the collision state C. Then, as shown in Fig. B.4, the transition probability matrix P of the Markov chain can be represented by

$$P = \begin{pmatrix} P_{II} & P_{IS} & P_{IC} \\ P_{SI} & 0 & 0 \\ P_{CI} & 0 & 0 \end{pmatrix}, \tag{B.2}$$

where P_{II} is the transition probability from state I to state I, and so on. According to [53], when a node leaves state I, which means the node begins a transmission, the node enters state S, if this transmission is successful. Otherwise, it enters state C. We assume that a

node will not emit another transmission immediately after the node finishes a transmission. Then, this implies that the node leaves state S or C, and then it transits to state I with probability one, i.e., $P_{SI} = P_{CI} = 1$. By using global balance equations and normalization condition, we have

$$P(I) = P(I)P_{II} + P(S) + P(C) = P(I)P_{II} + 1 - P(I), \tag{B.3}$$

from which we can straightforwardly obtain $P(I)$ by

$$P(I) = \frac{1}{2 - P_{II}} = \frac{1}{2 - (1 - p)}. \tag{B.4}$$

Note that a node leaves state I with probability p, and does not leave with probability $P_{II} = 1 - p$. The steady-state probability of states S and C can be expressed by

$$P(S) = P(I)P_{IS}, \tag{B.5}$$

and

$$P(C) = 1 - P(S) - P(I). \tag{B.6}$$

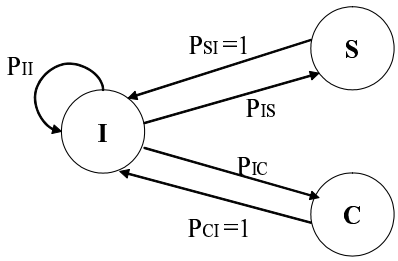


Figure B.4: The Markov chain model for a node.

We define the throughput TH by the fraction of time in which a node is engaged in successful data packet transmission. Then, TH can be expressed by

$$TH = \frac{P(S) \cdot T_{data}}{P(C) \cdot T_C + P(S) \cdot T_S + P(I) \cdot T_I}, \tag{B.7}$$

where T_S, T_I, and T_C are the durations of states S, I, and C.

For expressing P_S, we need to express P_{IS}, which is the probability from the idle state to the successful state. Let us denote by $P_{IS}(r)$ the transition probability from state I to state S, given that the distance between sending node (node A) and receiving node (node

B) is r as in Fig. B.2. Since we assume that the radius of hearing region of node A is one, P_{IS} can be expressed by

$$P_{IS} = \int_0^1 P_{IS}(r) f(r) dr, \tag{B.8}$$

where $f(r)$ is the probability density function of r $(0 < r < 1)$. Because nodes are uniformly distributed, we can obtain $f(r) = 2r$.

In the next three subsections, we will derive time (T_S, T_I, and T_C) and the probability $P_{IS}(r)$ at different states of IEEE 802.11, DCA, and BTMMAC.

B.1.3 The throughput of IEEE 802.11

Now firstly, we determine the transition probability of IEEE 802.11, P_{IS}. Following the four-way handshake of IEEE 802.11 as shown in Fig. B.2, $P_{IS}(r)$ is equal to the probability that node A transmits in a given time slot, node B does not transmit in the same time slot, and none of the nodes interfer with the handshakes. Then, $P_{IS}(r)$ can be expressed by

$$P_{IS}(r) = p \cdot (1-p) \cdot P_{rts} \cdot P_{cts} \cdot P_{data} \cdot P_{ack}, \tag{B.9}$$

where P_{rts} is the probability that no nodes interfere with the RTS reception, P_{cts} is the probability that no nodes interfere with the CTS reception, P_{data} is the probability that no nodes interfere with the data packet reception and P_{ack} interfere with the ACK reception.

As depicting in Fig. B.2 and using Eq. (B.1), P_{rts} is equal to the probability that no nodes in the region X transmit in the same time as node A does, and no nodes in the region Z transmit during $(T_{rts} + a + 1)$ period, we obtain

$$P_{rts} = \left\{ \sum_{i=0}^{\infty} (1-p)^i \cdot \frac{(\lambda X)^i}{i!} \cdot e^{-\lambda X} \right\} \times \left\{ \sum_{i=0}^{\infty} (1-p)^i \cdot \frac{(\lambda Z)^i}{i!} \cdot e^{-\lambda Z} \right\}^{T_{rts}+a+1}. \tag{B.10}$$

When node B hears the RTS correctly, it responses with a CTS. As illustrated in Fig. B.2, because the RTS was sent by node A that can block the nodes within the region X, node B may only be interfered by the nodes within the region Z during $(T_{cts} + a + 1)$ period. Therefore, the probability of P_{cts} can be expressed by

$$P_{cts} = \left\{ \sum_{i=0}^{\infty} (1-p)^i \cdot \frac{(\lambda Z)^i}{i!} \cdot e^{-\lambda Z} \right\}^{T_{cts}+a+1}. \tag{B.11}$$

After receiving CTS correctly, node A will send data packet to node B. From [55], we know that although RTS/CTS is exchanged successfully, it still has possibility that the collision happens in the period of data packet transmission. The probability of P_{data} is equal to

the probability that no nodes in the region Y transmit during $(T_{data} + a + 1)$, therefore we obtain

$$P_{data} = \left\{ \sum_{i=0}^{\infty} (1-p)^i \cdot \frac{(\lambda Y)^i}{i!} \cdot e^{-\lambda Y} \right\}^{T_{data}+a+1}. \tag{B.12}$$

After that, we consider that no nodes interfere with the reception of ACK and hence the probability of P_{ack} is 1. Therefore, using Eqs. (B.10), (B.11), and (B.12) in Eq. (B.9), $P_{IS}(r)$ of IEEE 802.11 can be expressed by

$$P_{IS}(r) = p \cdot (1-p) \cdot \{e^{-p\lambda X} e^{-p\lambda Z(T_{rts}+a+1)}\} \cdot \{e^{-p\lambda Z(T_{cts}+a+1)}\} \cdot \{e^{-p\lambda Y(T_{data}+a+1)}\} \cdot 1. \tag{B.13}$$

The handshake between the sender and the receiver may be interrupted during the collision period varies of from T_{rts} to $(T_{rts} + a + 1) + (T_{cts} + a + 1) + (T_{data} + a + 1)$. We consider the worst duration. Then, the duration of time slots of a node in the collision state of IEEE 802.11 is given by

$$T_C = T_{rts} + T_{cts} + T_{data} + 3a + 3, \tag{B.14}$$

and the duration in time slots of a node in the successful state of IEEE 802.11 is given by

$$T_S = (T_{rts} + a + 1) + (T_{cts} + a + 1) + (T_{data} + a + 1) + (T_{ack} + a + 1). \tag{B.15}$$

Using Eq. (B.8) in Eq. (B.5), $P(S)$ can be expressed. Using Eqs. (B.15), (B.4), (B.5), (B.6), and (B.14) in Eq. (B.7), the throughput of IEEE 802.11, TH_{IEEE}, can be expressed by

$$TH_{IEEE} = \frac{\frac{1}{1+p} P_{IS} \cdot T_{data}}{(1 - \frac{1}{1+p} - \frac{1}{1+p} P_{IS}) \cdot T_C + \frac{1}{1+p} P_{IS} \cdot T_S + \frac{1}{1+p} \cdot T_I}, \tag{B.16}$$

where T_I is equal to one slot time.

B.1.4 The throughput of DCA

Secondly, we determine the transition probability of DCA, P'_{IS}. Based on the condition for successful transmission given in Eq. (B.9) and using the same notation, P'_{IS} is equal to the probability that node A transmits in a given time slot on channel m, node B does not transmits in the same time slot on channel m, and none of the nodes interfere with the handshake. Then, $P'_{IS}(r)$ can be expressed by

$$P'_{IS}(r) = \sum_{m=1}^{M} \frac{p}{M} \cdot \left(1 - \frac{p}{M}\right) \cdot P'_{rts} \cdot P'_{cts} \cdot P'_{data} \cdot P'_{ack}. \tag{B.17}$$

Since DCA uses one control channel to exchange RTS/CTS messages, the collision of RTS/CTS messages is relatively reduced. The probability of $P'_{rts} \cdot P'_{cts}$ is equal to the

probability that no nodes in the region $B(r)$ transmit in the same time $(a+1)$ as node A does on channel m

$$P'_{rts} \cdot P'_{cts} = \sum_{i=0}^{\infty} \left(1 - \frac{p}{M}\right)^i \cdot \frac{(\lambda N(B))^i}{i!} \cdot e^{-\lambda N(B)}. \tag{B.18}$$

However, the collision still happens in the period of data packet transmission due to the hidden terminal problem as shown in Chapter 2. The probability of P'_{data} on channel m is expressed by

$$P'_{data} = \left\{ \sum_{i=0}^{\infty} \left(1 - \frac{p}{M}\right)^i \cdot \frac{(\lambda Y)^i}{i!} \cdot e^{-\lambda Y} \right\}^{T_{data}+a+1}. \tag{B.19}$$

We also consider that the probability of P'_{ack} is 1. Using Eqs. (B.18) and (B.19) in Eq. B.17, $P'_{IS}(r)$ of DCA can be expressed by

$$P'_{IS}(r) = M \cdot \frac{p}{M} \cdot \left(1 - \frac{p}{M}\right) \cdot \{e^{-\frac{p}{M}\lambda N(B)}\} \cdot \{e^{-\frac{p}{M}\lambda Y(T_{data}+a+1)}\} \cdot 1. \tag{B.20}$$

According to our analysis, the worst duration in time slots of a node in the collision state of DCA is given by

$$T'_C = T_{rts} + T_{cts} + T_{data} + 3a + 3, \tag{B.21}$$

and the duration in time slots of a node in the successful state of DCA is given by

$$T'_S = (T_{rts} + a + 1) + (T_{cts} + a + 1) + (T_{data} + a + 1) + (T_{ack} + a + 1). \tag{B.22}$$

Using Eq. (B.8) (substitute $P'_{IS}(r)$ for $P_{IS}(r)$) in Eq. (B.5), $P'(S)$ can be expressed. Using Eqs. (B.4), (B.5), (B.6), (B.21), and (B.22) in Eq. (B.7)), the throughput of DCA can be expressed by

$$TH_{DCA} = \frac{\frac{1}{1+p} P'_{IS} \cdot T_{data}}{(1 - \frac{1}{1+p} - \frac{1}{1+p} P'_{IS}) \cdot T'_C + \frac{1}{1+p} P'_{IS} \cdot T'_S + \frac{1}{1+p} \cdot T'_I}, \tag{B.23}$$

where T'_I is equal to one slot time.

B.1.5 The throughput of BTMMAC

Finally, we determine the transition probability of BTMMAC, P''_{IS}. We denoted by P_{bt} the probability that no nodes interfere with the Busy-Tone reception. Substituting P_{bt} for P'_{cts} in Eq. (B.17), we have

$$P''_{IS}(r) = \sum_{m=1}^{M} \frac{p_t}{M} \cdot \left(1 - \frac{p_t}{M}\right) \cdot P''_{rts} \cdot P_{bt} \cdot P''_{data} \cdot P''_{ack}. \tag{B.24}$$

By sensing on the busy tone, any other potential nodes defer to send packets in BTMMAC. Therefore, we have

$$P_{rts}^{''} \cdot P_{bt} = \sum_{i=0}^{\infty} \left(1 - \frac{p_t}{M}\right)^i \cdot \frac{(\lambda N(B))^i}{i!} \cdot e^{-\lambda N(B)}, \tag{B.25}$$

and

$$P_{data}^{''} = P_{ack}^{''} = 1. \tag{B.26}$$

As a consequence, $P_{IS}^{''}(r)$ can be expressed by

$$P_{IS}^{''}(r) = M \cdot \frac{p}{M} \cdot \left(1 - \frac{p}{M}\right) \cdot \left\{e^{-\frac{p}{M}\lambda N(B)(a+1)}\right\} \cdot 1 \cdot 1. \tag{B.27}$$

If we consider the worst duration in time slots of a node in the collision state of BTMMAC, then $T_C^{''}$ is given by

$$T_C^{''} = (T_{rts} + a + 1) + (T_{bt} + a + a), \tag{B.28}$$

and the duration in time slots of a node in the successful state of BTMMAC is

$$T_S^{''} = (T_{rts} + a + 1) + (T_{bt} + a + a) + (T_{data} + a + 1) + (T_{bt} + a + a). \tag{B.29}$$

Using Eq. (B.8) (substitute $P_{IS}^{''}(r)$ for $P_{IS}(r)$) in Eq. (B.5), $P^{''}(S)$ can be expressed. Using Eqs. (B.4), (B.5), (B.6), (B.28), and (B.29) in Eq. (B.7), the throughput of BTMMAC can be expressed by

$$TH_{BTMMAC} = \frac{\frac{1}{1+p} P_{IS}^{''} \cdot T_{data}}{(1 - \frac{1}{1+p} - \frac{1}{1+p} P_{IS}^{''}) \cdot T_C^{''} + \frac{1}{1+p} P_{IS}^{''} \cdot T_S^{''} + \frac{1}{1+p} \cdot T_I^{''}}, \tag{B.30}$$

where $T_I^{''}$ is equal to two slots time.

B.2 Numerical results

B.2.1 The hidden terminal problem

By using the Markov chain model, we compare the throughput of BTMMAC with DCA and IEEE 802.11 for different values of p at $a = 0.01$ and $M = 3$.

In Fig. B.5, we can find that the protocol using multiple channels has a better performance than using single channel. The reason is that the protocol using multiple channels has more spatial reuse capability than the protocol using single channel, which can reduce the collision relatively. Therefore, the protocol using multiple channels becomes more effective than using single channel. From the same figure, we can also find that BTMMAC has

a higher performance than DCA. The reason for this is BTMMAC allows nodes to sense channels by busy tone before they choose a channel. Therefore, the busy tone assisted multi-channel MAC protocol can alleviate the hidden terminal problem.

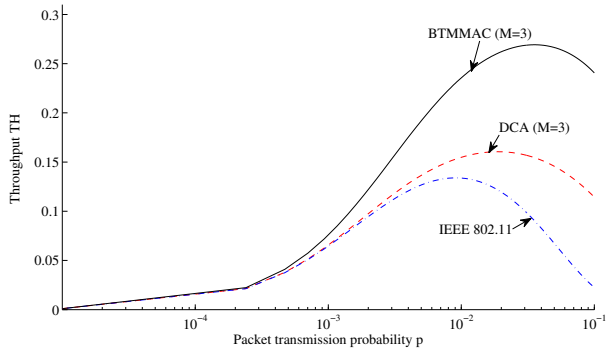


Figure B.5: The throughput of IEEE 802.11, DCA, and BTMMAC for $a = 0.01$ and $M = 3$

.

Appendix C

The payoff of n-player and m-channel stochastic channel-selection games

Let us consider n-player and m-channel stochastic channel-selection game using C_1, and the mixed-strategy $\sigma_i^*(s_{i,ch,ch'}) = 1/[m \cdot (m-1)]$ for $i \in \{1, 2, \ldots, n\}$, $\forall ch \in CH$ and $\forall ch' \in CH$ with $ch \neq ch'$. Note that the expected payoffs $\bar{u}_i(s_i, \sigma_{-i}^*)$ take the same value for all $s_i \in \{s_{i,ch_1,ch_2}, \ldots, s_{i,ch_1,ch_m}, \ldots, s_{i,ch_m,ch_{m-1}}\}$ at a totally mixed-strategy NE for each player $i \in \{1, 2, \ldots, n\}$. Furthermore, it is clear that the expected payoffs $\bar{u}_i(s_i, \sigma_{-i}^*)$ for all players are the same by symmetry, and we denote the value by q_n.

For each player, it should be noted that the expected payoff is composed of the sum of the expected payoff in initial and strategic phases, and both phases give the same payoff. With an eye to the mixed-strategy $\sigma_i^*(s_{i,ch,ch'}) = 1/[m \cdot (m-1)]$, we can compute the expected payoff given by Eq. (3.4) and rewrite q_n as

$$q_n = 2\left(\frac{r_n}{[m \cdot (m-1)]^{n-1}} - c\right),$$

where r_n is the sum of the reward, i.e., payoff without cost, in initial or strategic phases for $n \geq 2$.

Let us consider r_n. In case of two-player stochastic channel-selection game using C_1, we can see that there are $(m-1)^2$ combinations of successful channels and $(m-1)$ combinations of unsuccessful channels in both initial and strategic phases. Therefore, it can be computed that the sum of the reward for two-player stochastic channel-selection game using C_1 as $r_2 = (+1) \times (m-1)^2 + (-1) \times (m-1)$.

In case of more than two players, let us consider two players arbitrarily chosen among n players, and fix a strategy $s \in \{s_{i,ch_1,ch_2}, \ldots, s_{i,ch_1,ch_m}, \ldots, s_{i,ch_m,ch_{m-1}}\}$ of one player. Then, $(m-1)^2$ strategies of the other player result in successful channels, and the remaining $(m-1)$ strategies give unsuccessful channels. For each strategy in successful channel, the sum of reward is the same and given by r_{n-1} because it can be viewed as $(n-1)$-player stochastic channel-selection game. For the $(m-1)$ strategies resulting in unsuccessful channels, the

reward is given by -1 that is irrespective of the strategies of the other $(n-2)$ players. Therefore, we have the relation

$$r_n = (m-1)^2 \cdot r_{n-1} + (-1) \times (m-1) \cdot [m \cdot (m-1)]^{n-2},$$

and the solution is given by $r_n = 2 \cdot (m-1)^{2(n-1)} - [m \cdot (m-1)]^{n-1}$ for $n \geq 2$. It implies that the expected payoff q_n for n-player game using C_1 is given by

$$q_n = 4\left(\frac{m-1}{m}\right)^{n-1} - 2 - 2c,$$

if the mixed-strategy $\sigma_i^*(s_{i,ch,ch'}) = 1/[m \cdot (m-1)]$ for $i \in \{1, 2, \ldots, n\}$, $\forall ch \in CH$ and $\forall ch' \in CH$ with $ch \neq ch'$ is chosen by each player. By using Proposition 1, we can see that the mixed-strategy $\sigma_i^*(s_{i,ch,ch'}) = 1/[m \cdot (m-1)]$ is a NE for n-player and m-channel stochastic channel-selection game using C_1.

By using the same way, we can show that the mixed-strategy $\sigma_i^*(s_{i,ch,ch'}) = 1/m^2$ for $i \in \{1, 2, \ldots, n\}$, $\forall ch \in CH$ and $\forall ch' \in CH$ gives a NE of n-player and m-channel stochastic channel-selection game using C_2. It can be shown by elementary computation that the expected payoff of n-player and m-channel stochastic channel-selection game using C_2 is the same to that of n-player stochastic and m-channel channel-selection game using C_1.

Appendix D

The payoff of 2-player and m-channel using CRCS+

Let us consider 2-player and m-channel stochastic channel-selection game using C_3. For each player, it should be noted that the expected payoff is composed of the sum of the expected payoff in initial and strategic phases, and both phases give the same payoff. With an eye to the mixed-strategy $\sigma_i^*(s_{i,ch,ch}) = 1/[m^2 \cdot (m-1)]$ and $\forall ch \in CH$, we can compute the expected payoff given by Eq. (3.4) and rewrite q_2 as

$$q_2 = 2\left(\frac{r_2}{[m^2 \cdot (m-1)]^{n-1}} - c\right),$$

where r_2 is the sum of the reward.

In case of two-player stochastic channel-selection game using C_3, we can see that there are $m^3 - 2m^2 + m + 1$ combinations of successful channels and $m^2 - m - 1$ combinations of unsuccessful channels in both initial and strategic phases. Therefore, it can be computed that the sum of the reward for two-player stochastic channel-selection game using C_3 as $r_2 = (+1) \times (m^3 - 2m^2 + m + 1) + (-1) \times (m^2 - m - 1)$.

It implies that the expected payoff q_n for n-player game using C_3 is given by

$$q_n = 4\left(\frac{m-1}{m}\right) - 2 - 2c + \frac{2}{m^2(m-1)},$$

if the mixed-strategy $\sigma_i^*(s_{i,ch,ch}) = 1/[m \cdot (m-1)]$ for $i \in \{1, 2\}$ and $\forall ch \in CH$ is chosen by each player. By using Proposition 1, we can see that the mixed-strategy $\sigma_i^*(s_{i,ch,ch'}) = 1/[m^2 \cdot (m-1)]$ is a NE for 2-player and m-channel stochastic channel-selection game using C_3.

Bibliography

[1] B.-O. Hara and A. Petrick, "IEEE 802.11 handbook," A Designer's Companion, 1999.

[2] N. Shacham and J.-B. King, "Architectures and performance of multichannel multihop packet radio networks," IEEE Journal on Selected Areas in Communications, vol. 5, no. 6, pp. 1013–1025, Jul. 1987.

[3] R. Maheshwari, H. Gupta, and S.-R. Das, "Multichannel MAC protocols for wireless networks," Proc. 3rd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, vol. 2, pp. 393–401, Sep. 2006.

[4] S.-L. Wu, C.-Y. Lin, Y.-C. Seng, and J.-P. Sheu, "A new multi-channel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc network," Proc. 2000 International Symposium on Parallel Architectures, Algorithms and Networks, pp. 232–237, Dec. 2000.

[5] A. Nasipuri, J. Zhuang and S.-R. Das, "A multichannel CSMA MAC protocol for multi-hop wireless networks," Proc. IEEE Wireless Communications Networking Conference, Sep 1999.

[6] A. Nasipuri, and S.-R. Das, "Multi-channel CSMA with signal power-based channel selection for multi-hop wireless networks," Proc. IEEE Vehicular Technology Conference, Sep. 2000.

[7] S.-L. Wu, C.-Y. Lin, Y.-C. Seng, and C.-Y. Lin, "A multi-channel MAC protocol with power control for multi-hop mobile ad-hoc networks," The Computer Journal, vol. 45, no. 1, pp. 101–110, 2002.

[8] W.-C. Hung, K.-E. Law, and A.-L. Garcia, "A dynamic multi-channel MAC for ad-hoc LAN," Proc. 21st Biennial Symposium on Communications, pp. 31–35, Jun. 2002.

[9] P.-J. Wu and C.-N. Lee, "Connection-oriented multi-channel MAC protocol for ad-hoc networks," Computer Communications, vol. 32, pp. 169–178, 2009.

[10] C.-W. Chen, C.-C. Weng, and C.-J. Ku, "An overlapping and pipelining data transmission MAC protocol," Parallel Computing, vol. 35, no. 6, pp. 313-330, Jun. 2009.

[11] R.-S. Huang, H.-Q. Zhai, C. Zhang, and Y.-G. Fang, "SAM-MAC: An efficient channel assignment scheme for multi-channel ad hoc networks," Computer Networks, vol. 52, no. 8, pp. 1634–1646, Jun. 2008.

[12] I. Wormsbecker and C. Williamson, "On channel selection strategies for multi-channel MAC protocols in wireless ad hoc networks," Proc. 2006 Wireless and Mobile Computing, Networking and Communications, pp. 212–220, June 2006.

[13] J.-P. Shi, T. Salonidis, and E.-W. Knightly, "Starvation mitigation through multi-channel coordination in CSMA multi-hop wireless networks," Proc. 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 214–225, May 2006.

[14] T. Luo, M. Motani, and V. Srinivasan, "CAM-MAC: A cooperative asynchronous multi-channel MAC protocol for ad hoc networks," Proc. 2006 Broadband Communications, Networks and Systems, pp. 1–10, Oct. 2006.

[15] J. So and N. Vaidya, "Multi-channel MAC for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver," Proc. 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 222–233, May 2004.

[16] J. Wang, Y. Fang, and D. Wu, "A power-saving multi-radio multi-channel MAC protocol for wireless local area networks," Proc. 25th IEEE International Conference on Computer Communication, pp. 1–12, Apr. 2006.

[17] J. Zhang, G. Zhou, C. Huang, S.-H. Son, and J.-A. Stankovic, "TMMAC: An energy efficient multi-channel MAC protocol for ad hoc networks," Proc. IEEE International Conference on Communications, pp. 3554–3561, Jun. 2007.

[18] G.-J. Yu and C.-Y .Chang, "An efficient cluster-based multi-channel management protocol for wireless ad hoc networks," Computer Communications, vol. 30, no. 8, pp. 1742–1753, Jun. 2007.

[19] A. Tzamaloukas and J. Garcia-Luna-Aceves, "Channel-hopping multiple access," Proc. IEEE International Conference on Communications, 2000.

[20] J. Chen, S. Sheu, and C. Yang, "A new multichannel access protocol for IEEE 802.11 ad hoc wireless LANs," Proc. Personal, Indoor and Mobile Radio Communications, vol. 3, pp. 2291–2296, Sep. 2003.

[21] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad hoc wireless networks," Proc. 10th Annual International Conference on Mobile Computing and Networking, pp. 216–230, Sep. 2004.

[22] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multi-hop wireless ad hoc networks," IEEE Communications Magazine, vol. 39, no. 6, pp. 130–137, Jun. 2001.

[23] S. Xu and T. Saadawi, "Performance evaluation of TCP algorithms in multi-hop wireless packet networks," Journal of Wireless Communications and Mobile Computing, vol. 2, no. 1, pp. 85–100, 2002.

[24] G. Holland and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," Wireless Networks, vol. 8, no. 2-3, pp. 275–288, 2002.

[25] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP fairness in ad hoc wireless networks using neighborhood red," Proc. 9th Annual International Conference on Mobile Computing and Networking, pp. 16–28, Sep. 2003.

[26] L. Yang, W. Seah, and Q. Yin, "Improving fairness among TCP flows crossing wireless ad hoc and wired networks," Proc. 9th Annual International Conference on Mobile Computing and Networking, pp. 57–63, Sep. 2003.

[27] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multi-hop wireless channel on TCP throughput and loss," Proc. 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, pp. 1744–1753, 2003.

[28] T. Kuang and C. Williamson, "A bidirectional multi-channel MAC protocol for improving TCP performance on multi-hop wireless ad hoc networks," Proc. 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp. 301–310, Oct. 2004.

[29] J. N. Webb, Game Theory: Decisions, Interaction and Evolution, Springer, 2006.

[30] S. Xu and T. Saadawi, "Revealing and solving the TCP instability problem in 802.11 based multi-hop mobile ad-hoc networks," Proc. 54th IEEE Vehicular Technology Conference, vol. 1, pp. 257–261, 2001.

[31] S. Xu and T. Saadawi, "Revealing TCP unfairness behavior in 802.11 based wireless multi-hop networks," Proc. 12th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, vol. 2, pp. E-83–E-87, 2001.

[32] K. Chen, Y. Xue, S. Shah, and K. Nahrstedt, "Understanding bandwidth-delay product in mobile ad hoc networks," Computer Communications, vol. 27, no. 10, pp. 923–934, Jun. 2004.

[33] B. Han and K. Kawanishi, "Conditionally randomized channel selection algorithm for multi-channel MAC protocol in ad hoc networks," IEICE Transactions on Communications, vol. E94-B, no. 4, pp. 940–950, April 2011.

[34] A. Nasipuri, and J. Mondhe, "Multi-channel MAC with dynamic channel selection for ad hoc networks," Jan. 2004. *http://www.ece.uncc.edu/∼anasipur/pubs/MC-04.pdf*

[35] The network simulator ns-2.
Web site: *http://www.isi.edu/nsnam/ns/.*

[36] R. K. Jain, "The Art of Computer Systems Performance Analysis," John Wiley and Sons, April 1991.

[37] B. Han and K. Kawanishi, "Using game theory to investigate stochastic channel selection for multi-channel MAC protocol," Proc. 13th International Conference on Communication Systems IEEE ICCS, 2012.

[38] A. Amer and F. Gebali "General model for single and multiple channels WLANs with quality of service support," International Journal of Wireless & Mobile Networks, vol. 1, no. 2, pp. 1–19, Nov. 2009.

[39] P. Mahonen, J. Riihijarvi, and M. Petrova, "Automatic channel allocation for small wireless local area networks using graph coloring algorithm approach," Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pp. 536–539, Sep. 2004.

[40] P. Akl and A. Arepally, "Dynamic channel assignment in IEEE 802.11 networks," Proc. IEEE International Conference on Portable Information Devices, pp. 1–5, May 2007.

[41] M. Felegyhaz, M. Cagalji, S.-S. Bidokhti, and J.-P. Hubaux. "Non-cooperative multi-radio channel allocation in wireless networks," Proc. 26th IEEE International Conference on Computer Communications, pp. 1442–1450, May 2007.

[42] F. Wu, S. Zhong, and C. Qiao. "Globally optimal channel assignment for non-cooperative wireless networks," Proc. 27th IEEE International Conference on Computer Communications, pp. 1543–1551, April 2008.

[43] L. Gao and X. Wang. "A game approach for multi-channel allocation in multi-hop wireless networks," Proc. 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 303–312, May 2008.

[44] L. Gao, X. Wang, and Y.-Y. Xu. "Multi-radio channel allocation in multi-hop wireless networks," IEEE Transactions on Mobile Computing, vol. 8, no. 11, pp. 1454–1468, Nov. 2009.

[45] R. B. Myerson, Game Theory: Analysis of Conflict, Cambridge, MA: Harvard Univ. Press, 1960.

[46] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," IEEE Journal on Selected Areas in Communications, vol. 18, no. 3, pp. 535–547, March 2000.

[47] B. Han, K. Kawanishi, and Y. Onozato "Congestion window limit multi-channel MAC protocol: Employing random channel selection strategy for multi-hop ad hoc networks," Proc. 12th International Symposium on Wireless Personal Multimedia Communications (WPMC'09), Hotel Metropolitan Sendai, Sendai, Japan, S40: MAC (USB memory), Sep. 2009

[48] M. Gerla, K. Tang, and R. Bagrodia, "TCP performance in wireless multi-hop networks," Proc. 2nd IEEE Workshop on Mobile Computer Systems and Applications, pp. 41–50, 1999.

[49] S. Floyd and T. Henderson, "The NewReno modification to TCP's fast recover algorithm," RFC 2582, 1999.

[50] D.-B. Johnson and D.-A. Maltz, "Dynamic source routing in ad hoc wireless networks," Mobile Computing, vol. 353, pp. 153–181,1996.

[51] K. Xu, M. Gerla, and S. Bae, "Effectiveness of RTS/CTS handshake in IEEE 802.11 based ad hoc networks," Ad hoc Networks, vol. 1, pp. 107–123, 2003.

[52] B. Han and K. Kawanishi, "A Busy-Tone based MAC protocol for ad hoc networks using multiple channels," Proc. The Queuing Symposium Stochastic Models and Their Applications, 2009.

[53] L. Wu and P.-K. Varshney, "Performance analysis of CSMA and BTMA protocols in multi-hop networks," Information Sciences, vol. 120, pp. 159–177, 1999.

[54] H. Takagi and L. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminal," IEEE Transactions on Communications, vol. 3, pp. 246–257, 1984.

[55] R. Wei, "Hidden terminal analysis on TCP stability over multi-hops ad hoc networks," Computer Technology and Application, vol. 1, pp. 14–28, 2007.