WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2015; **15**:1015–1025
Published online 7 July 2014 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.2496

SPECIAL ISSUE PAPER

An adaptive multichannel protocol for large-scale machine-to-machine networks

Chi-Hsien Yen[1,2*], Chen-Yu Hsu[1,2] and Chun-Ting Chou[2,3]

[1] Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan
[2] Intel–NTU Connected Context Computing Center, Taipei, Taiwan
[3] Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan

ABSTRACT

With the emergence of Internet of Things (IoT), trillions of devices will soon be interconnected and provide new services. The success of IoT relies on the scalability of underlying machine-to-machine communication networks. In this paper, we propose a distributed and adaptive multichannel protocol to address the scalability issue. The proposed protocol is composed of (i) real-time estimation of competing devices, (ii) adaptive channel access probability, and (iii) asynchronous resource reservation. The three features ensure that channel utilization is maximized even when the number of competing devices and their traffic fluctuate dramatically. Our numerical and simulation results show that the proposed protocol can achieve a channel utilization of up to 93%, especially when the number of competing machine is large. Copyright © 2014 John Wiley & Sons, Ltd.

KEYWORDS
M2M; medium access control; common control channel

***Correspondence**
Chi-Hsien Yen, Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan.
E-mail: b98901112@ntu.edu.tw

1. INTRODUCTION

Internet of Things (IoT) is considered as the most important evolution for the Internet in the post-smartphones era. Through the IoT, trillions of machines will be interconnected and provide new applications and services. The key to interconnect such a huge amount of machines is the scalability of the machine-to-machine (M2M) communication networks and protocols. Any good solution for M2M communication must scale well with the number of machines so as to transport the unprecedented amount of data.

Multichannel operation is a promising solution for M2M communication given that machines could transmit concurrently on different channels. Many existing wireless communication protocols such as IEEE 802.11 (i.e., WiFi) and 802.15.4 (i.e., ZigBee) have adopted a multichannel architecture. However, these protocols do not fully utilize multichannel operation in the sense that machines do not switch among channels on a regular basis. For example, a WiFi access point (AP) usually selects a channel (e.g., channel 1, 6, or 11) and communicates with its stations using that channel. Although different APs may use different channels, the overall channel utilization may not be balanced and maximized.

To make best use of multichannel operation, each device should use all channels in a dynamic manner depending on the channel loads and the number of competing devices. Such multichannel operation can be realized in a centralized or distributed network. In a centralized network, a controller allocates channel resources to competing machines. The presence of a controller simplifies the process of resource allocation. One major problem of the centralized solution is signaling overhead. When the number of machines is large, a significant amount of resource/time will be spent for scheduling requests and responses. A centralized network is also subject to the single node (i.e., the controller) failure problem. In a distributed network, machines negotiate with each other for channel access. Depending on how the negotiation is carried out, distributed multichannel networks can be further classified as channel hopping-based or common control channel (CCC)-based.

In a hopping-based network, machines switch among different channels by following specially designed hopping sequences. When machines hop to the same channel at the same time, their communication can start. The advantage of a hopping-based solution is its low signaling overhead. The key challenge is the design of hopping sequences to guarantee "encounters" between any two

machines. Although many different sequences were developed to provide such guarantees [1–5], timely encounters usually depends on the number of channels in a network. The more channels in a network, the longer two machines need to wait between consecutive encounters. If machines operate at a low duty cycle (as many machines will do in the IoT), it is even more difficult for them to encounter and communicate with each other on time.

In a CCC-based network, one of the channels is designated as the control channel. On this control channel, machines negotiate with each other and reserve channels for data transmission. Because negotiation is made on a fixed channel, the encounter time is independent of the number of channels in a network. Many CCC-based protocols have been proposed for distributed multichannel networks. In [6,7], a dedicate control channel protocol was proposed. A machine using this protocol needs to be equipped with two independent transceivers. One of the transceivers is locked onto the control channel for channel reservation while the other is tuned to different channels for data transmission. By doing so, data transmission and negotiation can proceed concurrently, and channels can be utilized more efficiently. The drawback is that the hardware is more complicated and machines consume more power because of the use of dual transceivers.

In order to relax the requirement of dual transceivers, a split-phase protocol was then proposed [8]. In the split-phase protocol, time is divided into periodical intervals. Each interval is further divided into a negotiation phase and a data transmission phase. During the negotiation phase, machines stay in the control channel for resource reservation. Machines that complete their reservation start their data transmission at the end of negotiation phase. Given that only one channel is utilized during the negotiation phase, the overall utilization could be severely affected. In [9], the authors showed that the utilization is very sensitive to the length of the negotiation phase T_n in a split-phase protocol. They concluded that the length of the data transmission phase has little impact on the selection of T_n based on the assumption that all machines have no buffer space. However, how to determine the optimal T_n was not addressed. In [10], the authors attempted to improve the split-phase protocol using dynamic adjustment of T_n. The adjustment algorithm is simple. A machine broadcasts a request for (i) increasing T_n after an unsuccessful negotiation or (ii) decreasing T_n after an incomplete data transmission. On the basis of the collected requests, machines increase or decrease T_n by one time unit using the majority rule. Obviously, such heuristic adjustment cannot minimize the waste in the negotiation phase.

In this paper, we attempt to design an efficient CCC-based protocol for a large-scale M2M network. The design challenge here is that the number of machines that reserve channels in every negotiation phase is a random number, due mainly to the sporadic or bursty nature of M2M traffic. A design based on the worst-case scenario (i.e., assuming that all machines always have data to transmit) obviously does not work because a large T_n will be used and result in poor utilization. In order to solve the problem, we first develop a "baseline" split-phase protocol. Machines using the baseline protocol estimate the number of competing machines before a negotiation phase starts. On the basis of the estimation result, the machines determine T_n and an access probability p. The access probability determines how aggressively machines negotiate with each other during the negotiation phase. A mathematical model is then developed to select T_n and p such that the channel utilization can be maximized. We then introduce two mechanisms to enable asynchronous resource reservation in our baseline protocol. As we will show in this paper, asynchronous reservation fully utilizes the resource wasted by a split-phase protocol. Our numerical and simulation results show that the proposed protocol outperforms the existing split-phase protocols, especially when the number of competing devices is large.

The rest of this paper is organized as follows. In Section 2, we introduce the system settings and assumptions used in our protocol design. In Section 3, we first valuate the impact of T_n and p on the channel utilization of a split-phase protocol. Our baseline protocol is then presented, and the mathematical models for determining optimal T_n and p are also discussed. The numerical results and performance evaluation are given in Section 4. In Section 5, we present the two mechanisms that enable asynchronous reservation. Finally, the paper is concluded in Section 6.

2. SYSTEM SETTINGS AND ASSUMPTIONS FOR THE BASELINE SPLIT-PHASE PROTOCOL

In this paper, we consider an M2M network with N non-overlapping channels. Time is divided into periodic intervals with a fixed length of T_{total}. T_{total} is usually determined by the delay upper bound of M2M applications. Each interval is further divided into an estimation phase T_e, negotiation phase T_n, and data transmission phase T_d as shown in Figure 1. Time is slotted in the first two phases. During the estimation phase, each machine estimates the number of machines, M, that intend to transmit in the upcoming data transmission phase. In this paper, M is assumed to be a random number given the sporadic or bursty nature of M2M traffic.

In the negotiation phase, machines negotiate with each other by exchanging request and reply messages. The request message is transmitted at the beginning of each slot with a probability p. After having successfully received a request message, the receiver waits for one interframe space (with length of one slot) and sends back a reply message. After that, the rest of the machines wait for another interframe space to send their request messages with a probability p. If collision happens on the CCC, all machines also wait for one interframe space to resend their request messages with a probability p. The length of the request and reply messages, T_{req} and T_{rep}, are assumed to be 18 and 15 time slots, respectively. Each time slot is set

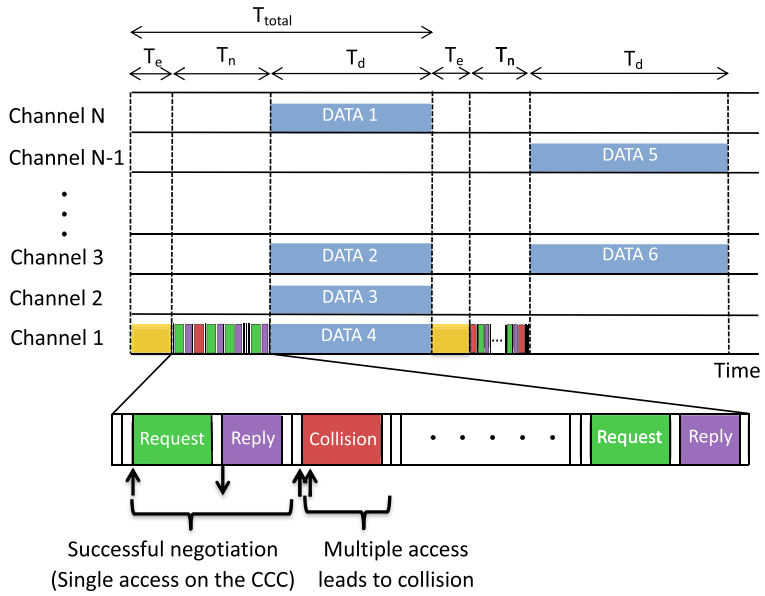


Figure 1. Timing structure of the proposed protocol.

to 20 μs. These values are chosen on the basis of the design of Request-to-Send (RTS) and Clear-to-Send (CTS) frames in the IEEE 802.11 protocol. The details of the negotiation process is also illustrated in Figure 1.

At the end of the negotiation phase, those successfully negotiated machines switch to the data channels reserved earlier and start data transmission. Throughout this paper, we assume that each machine is equipped with only one transceiver so as to better model low-power, low-complexity machines in M2M networks.

3. ADAPTIVE SPLIT-PHASE MULTICHANNEL PROTOCOL

In a split-phase multichannel protocol, the length of the negotiation phase, T_n, has a significant impact on the overall channel utilization. If T_n is too short, only a few machines can complete negotiation before the data transmission phase starts. As a result, many channels are left unused during the data transmission phase. If T_n is sufficiently long, all machines may complete negotiation. However, a longer T_n implies a shorter T_d given that T_{total} is a fixed value. Therefore, little time will be left for data transmission because data transmission cannot proceed concurrently with negotiation under our single-transceiver assumption. Such trade-off suggests that there exists an optimal T_n that maximizes the channel utilization. In what follows, we first investigate the impact of T_n on channel utilization. On the basis of our findings, an algorithm to determine T_n that maximizes the channel utilization will be developed.

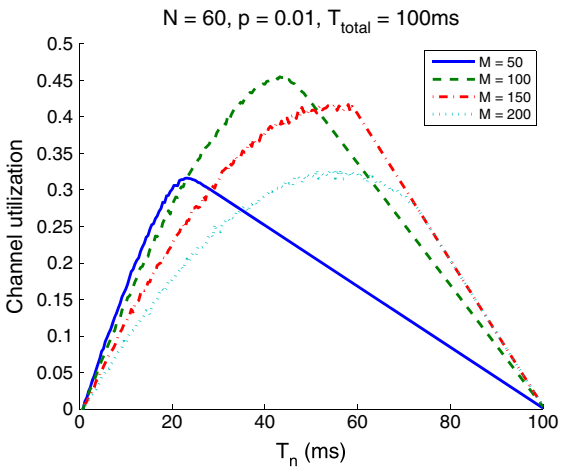


Figure 2. Channel utilization versus T_n: $T_{\text{total}} = 100$ ms and $p = 0.01$.

3.1. Impact of T_n and p on channel utilization

We first consider an M2M network with $N = 60$ channels. The number of machines that intend to negotiate, M, varies from 50 to 200. Figure 2 shows the channel utilization under different M's and T_n's. In this paper, channel utilization is defined as the ratio of channel time used for data transmission and is calculated by

$$U = \frac{T_d}{T_{\text{total}}} \times \frac{N_{\text{used}}}{N} \tag{1}$$

where N_{used} is the number of reserved channels for data transmission during T_d. Here, it is assumed that $T_e = 0$,

so we can focus on T_n's impact on channel utilization. The results show that (i) channel utilization varies significantly with T_n and (ii) there exists an optimal T_n that maximizes the channel utilization for any given M. In Figure 2, both the optimal T_n and the resulting maximum utilization varies with M. Because the optimal T_n varies with M, and M changes frequently in large-scale networks, it is infeasible to determine T_n off-line.

In Figure 2, we assume that the access probability in the negotiation phase, p, is fixed at 0.01. The value of p determines how aggressively machines contend for access during the negotiation phase and, consequently, also determines the number of machines that complete negotiation. Therefore, p also determines the overall channel utilization. Figure 3 shows the channel utilization under different p's for $T_n = 20$ ms and $N = 60$. The figure shows that the utilization is very sensitive to the value of p. In addition, there also exists an optimal p for given T_n and M. Again, the optimal p depends on M and therefore cannot be determined off-line. The study illustrates that dynamic adaptation of p and T_n according to M is the key to the efficiency of a split-phase multichannel network.

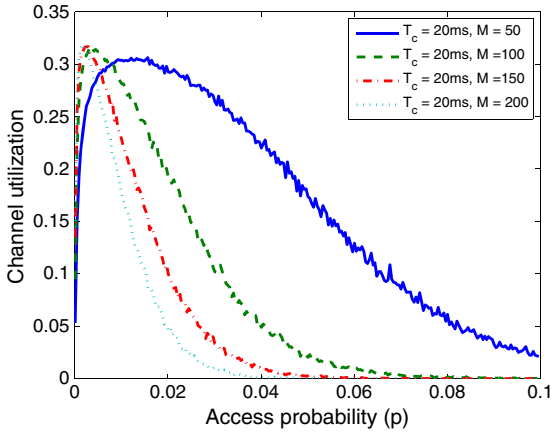


Figure 3. Channel utilization versus p: $T_{\text{total}} = 100$ ms and $T_n = 20$ ms.

3.2. Real-time estimation of *M*

In order to determine optimal T_n and p that maximize channel utilization, each machine must real-time estimate the value of M. In this paper, we propose a light-weight estimation algorithm. Our algorithm relies on so-called "busy tones" for machines to advertise their intention for negotiation. The basic idea of the proposed algorithm is similar to the solution in [11], but our solution focuses on using a small T_e to achieve a reasonable estimation of M. The details of the proposed algorithm is given as follows.

The proposed estimation algorithm is composed of two phases, including coarse phase and refine phase. In the coarse phase, each of the machines sends a busy tone on the CCC in the first time slot with a probability of $p_1 = 1/2$. If a machine sends a busy tone, it will send a busy tone in the second slot with a probability of $p_2 = \frac{1}{2^2}$. The process continues with $p_i = \frac{1}{2^i}$, where i is the index of slots in the negotiation phase. If a machine does not send a busy tone, it listens during the slot and determines whether or not some busy tones are detected. If a busy tone is received in slot i, the machine sends a busy tone with a probability of $p_{i+1} = \frac{1}{2^{i+1}}$ in slot $i+1$. Otherwise, the coarse phase is considered completed for the machine.

Figure 4 shows a sample result of the coarse phase. Here, we assume $M = 5$. The figure shows that all machines do not send busy tones in the third time slot. In this case, the coarse phase is completed in the third slot. Given that each machine halves the probability of sending busy tones

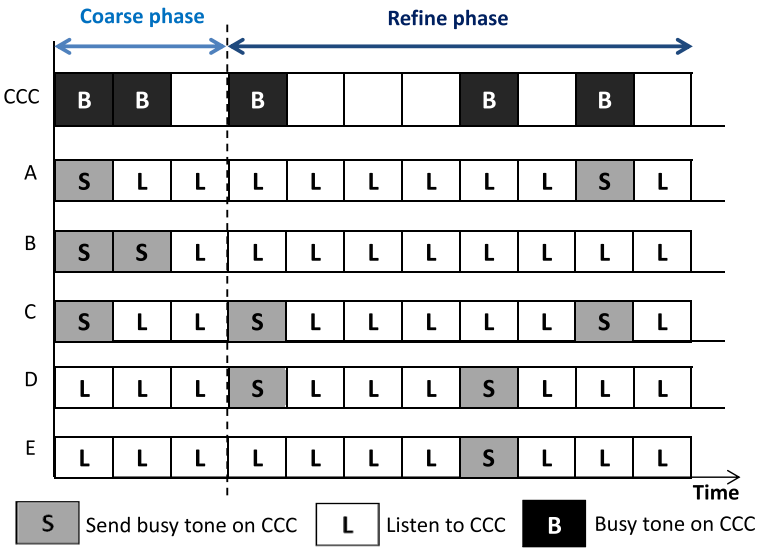


Figure 4. An example of the proposed estimation algorithm: $M = 5$.

Wirel. Commun. Mob. Comput. 2015; **15**:1015–1025 © 2014 John Wiley & Sons, Ltd.
DOI: 10.1002/wcm

in every time slot, the average length of the coarse phase is $\log_2 M$, which is acceptable especially when M is large.

In the refine phase, each machine sends a busy tone in every time slot with the probability used by the machine to send the last busy tone in the coarse phase. The length of the refine phase, L_r, is determined in advance, depending on the accuracy needed. On the basis of the extensive numerical results, we set L_r to 100 time slots. Such a value provides a reasonable result for larger M's. The expected value of the number of busy tones detected and sent by the machine (i.e., the total number of slots with busy tones on the CCC during the refine phase), B_r, could be calculated by

$$E[B_r] = L_r \times [1 - (1 - p_b)^M] \tag{2}$$

where p_b is the transmission probability of busy tones in each slot. On the basis of Equation (2), at the end of the refine phase, each machine estimates the number of machines that intend to negotiate in the negotiation phase by

$$\hat{M} = \frac{\log(1 - E[B_r]/L_r)}{\log(1 - p_b)} \tag{3}$$

An example of a refine phase with $p_b = 1/8$ and $L_r = 8$ is also shown in Figure 4. Machine A sends a busy tone in the seventh slot and detects busy tones in the first and fifth slots in the refine phase. Therefore, $B_r = 3$ and $\hat{M}$ is calculated as 3.52. Note that when M is large, B_r will be very close to $E[B_r]$ according to the law of large number and the central limit theorem.

Figure 5 shows the estimation results of the proposed algorithm. The figure shows the distribution of 10 000 estimated values of M when $M = 100$. The results show that the average of $\hat{M}$ is equal to M and the standard deviation is 17.4. The average total time for each estimation is 107 time slots, which are only about six request messages long. Consider that there are 100 machines that intend to send request and reply messages in the negotiation phase, such overhead (<5%) incurred by our estimation algorithm is negligible.

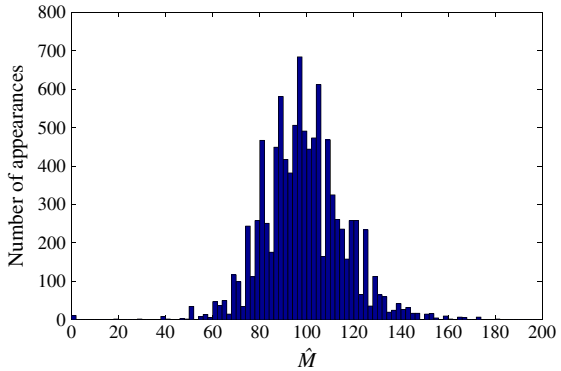


Figure 5. Distribution of $\hat{M}$: $M = 100$ and $L_r = 100$.

3.3. Optimal access probability p_{opt}

As we show in Section 3.1, there exists an optimal p that maximizes the number of machines that complete negotiation for a given T_n. In other words, there should exist an optimal p that minimizes the time needed to complete m pairs of negotiation. In this section, we derive such an optimal access probability first. Because a machine that completes its negotiation will not participate in the rest of the negotiation process, the number of negotiating machines will decrease gradually. The access probability that these remaining machines use might also change accordingly. Denote p_i as the access probability used by each machine when i machines are negotiating. Define T_i as the time slots needed for i machines to complete all negotiations. The expected value of T_i can be computed in a recursive way as

$$\begin{aligned} E[T_i] = {} & P_{i,1} \times \{1 + E[T_i]\} \\ & + P_{i,2}\{T_{\text{req}} + T_{\text{rep}} + 2 + E[T_{i-2}]\} \\ & + (1 - P_{i,1} - P_{i,2})\{T_{\text{req}} + 1 + E[T_i]\} \end{aligned} \tag{4}$$

The first term in Equation (4) represents the event that none of the machine accesses the channel in the first slot. As a result, one time slot is wasted, and the negotiation process restarts as if nothing happens. The probability of this event, $P_{i,1}$, can be obtained as $P_{i,1} = (1 - p_i)^i$. The second term represents the event that exactly one machine accesses the channel and thus completes negotiation with its targeted device. A total of $T_{\text{req}} + T_{\text{rep}} + 2$ is needed for the two machines to complete the negotiation, and $E[T_{i-2}]$ is needed for the rest of $i-2$ machines to complete their negotiation. The probability of this event, $P_{i,2}$, can be obtained as $P_{i,2} = \binom{i}{1} p_i (1-p_i)^{i-1}$. Finally, the third term represents the event that more than one machine access the channel and collide with each other. Therefore, $T_{\text{req}} + 1$ is wasted, and $E[T_i]$ is needed for these i machines to complete negotiation. The probability of this event can be obtained as $1 - P_{i,1} - P_{i,2}$.

Equation (4) can be simplified as

$$E[T_i] = T_{\text{rep}} + 1 + E[T_{i-2}] + \frac{T_{\text{req}} - T_{\text{req}}(1 - p_i)^i + 1}{i \times p_i (1 - p_i)^{i-1}}. \tag{5}$$

In Equation (4), p_i only appears in the last term. Therefore, the optimal p_i, $p_{i,\text{opt}}$, that minimizes $E[T_i]$ can be obtained by

$$p_{i,\text{opt}} = \arg\min_{p_i} \frac{T_{\text{req}} - T_{\text{req}}(1 - p_i)^i + 1}{i \times p_i (1 - p_i)^{i-1}}. \tag{6}$$

By simplifying Equation (6), we can calculate $p_{i,\text{opt}}$ for a given number of remaining machines i in the network.

3.4. Optimal negotiation phase, $T_{n,\text{opt}}$

It is observed from Section 3.1 that there exists an optimal T_n that maximizes channel utilization. In general, not all of M machines can complete negotiation within the optimal T_n. Take $M = 200$ in Figure 2 as an example. Channel utilization is maximized when $T_n = 56$ ms. Within such T_n, only 90 machines complete negotiation (i.e., $N_{\text{used}} = \frac{90}{2} = 45$). In this section, we attempt to find the optimal T_n when M machines intend to negotiate with each other. Assume that $2m$ out of M machines complete their negotiation in the optimal $T_{n,M,\text{opt}}$. According to Section 3.3, these M machines initially must use an access probability derived in Equation (6), $p_{M,\text{opt}}$. Once the first pair of machines complete their negotiation, the rest of $M-2$ machines initially must use an access probability equal to $p_{M-2,\text{opt}}$. The negotiation process continues until the mth pair of machines complete their negotiation (using an access probability equal to $p_{M-2m+2,\text{opt}}$).

In order to determine $T_{n,M,\text{opt}}$, we first define m_j as the number of machines that complete their negotiation in j time slots. m_j here is also a random variable. On the basis of Equation (1), $T_{n,M,\text{opt}}$ can be calculated by maximizing the expected channel utilization $E[U]$ as

$$\begin{aligned} T_{n,M,\text{opt}} &= \arg\max_{T_n} E[U] \\ &= \arg\max_{T_n} \begin{cases} \frac{T_d}{T_{\text{total}}} \times \frac{E[m_{T_n}]/2}{N}, & \text{if } \frac{E[m_{T_n}]}{2} < N \\ \frac{T_d}{T_{\text{total}}}, & \text{if } \frac{E[m_{T_n}]}{2} \geq N \end{cases} \end{aligned} \tag{7}$$

where $E[m_{T_n}]$ represents the expected value of the number of machines that complete their negotiation in T_n, and N_{used} in Equation (1) is replaced by $E[m_{T_n}]/2$. We assume that all m_{T_n} machines fully utilize the data transmission phase. Therefore, in the case of $\frac{E[m_{T_n}]}{2} \geq N$, we allow at most N pairs of machines to reserve data channels.

From the discussion in Section 3.3, the expected value of m_j, $E[m_j]$ can also be computed in a recursive way as

$$\begin{aligned} E[m_j] = {} & P_{m_j,1,\text{opt}} \times E[m_{j-1}] \\ & + P_{m_j,2,\text{opt}} \times \{E[m_{j-(T_{\text{req}}+T_{\text{rep}}+2)}] + 2\} \\ & + P_{m_j,3,\text{opt}} \times E[m_{j-(T_{\text{req}}+1)}] \end{aligned} \tag{8}$$

where $P_{m_j,1,\text{opt}}$, $P_{m_j,2,\text{opt}}$, and $P_{m_j,3,\text{opt}}$ represent the probabilities of the three events for channel access explained in Section 3.3, with i replaced by m_j and p_i replaced by $p_{m_j,\text{opt}}$ in Equation (6). $E[m_{T_n}]$ in Equation (7) can be calculated using Equation (8) with $j = T_n$. Finally, $T_{n,M,\text{opt}}$ can be obtained numerically using Equation (7).

4. NUMERICAL ANALYSIS AND EVALUATION

In this section, we compare the channel utilization of the proposed protocol with two other protocols denoted as OPTIMAL and FIX, respectively. OPTIMAL is an ideal protocol that knows M without estimation and uses the optimal T_n and T_d derived in our paper. OPTIMAL is used as the benchmark to evaluate the performance of the proposed protocol. On the other hand, FIX does not real-time estimate M and can only use fixed T_n and p. In our simulation, T_n in FIX is set as 20% of T_{total}, according to the protocol proposed by So and Vaidya [8] to achieve "good" performance.

For all of the simulations, the number of channels N is 40, and the length of T_{total} is 100 ms (i.e., 5000 time slots). The number of machines, M, follows a uniform distribution and changes for each T_{total}. Denote $\bar{M}$ as the mean of M, σ as the standard deviation of M, and $\mathcal{M}$ as the sample space of M. We have $\mathcal{M} = [\bar{M} - a, \bar{M} + a]$, and $\sigma = \sqrt{\frac{((2a+1)^2-1)}{12}}$, where σ is the standard deviation of M. In our simulations, the parameter a is varied to investigate the impact of σ on the proposed protocol.

4.1. The impact of $\bar{M}$ on U

Figure 6 shows the channel utilization under different $\bar{M}$'s in the three protocols. Here, a is fixed at 10 so that we can focus on the impact of $\bar{M}$. The results show that the utilization in the proposed protocol is very close to that in OPTIMAL. This result verifies that the estimation phase in our protocol incurs very little overhead. Our further analysis indicates that the length of the estimation phase, T_e, is about 107 slots on average, which is only $107/5000 \simeq 2\%$ of the total duration. In addition, the channel utilization in the proposed protocol increases with $\bar{M}$ until $\bar{M} \approx 80 (= 2 \times N)$. In our protocol, p is adjusted by each device according to $\hat{M}$ so that the efficiency of negotiation during T_n is almost unaffected as $\bar{M}$ increases. When $\bar{M} > 80$, the proposed protocol automatically disallows more than 80 devices to complete negotiation in T_n by adjusting T_n accordingly. With such distributed adaption, the channel utilization does not degrade as in FIX when $\bar{M}$ is large. In FIX, the number of devices is not estimated so that p has to be fixed. Three different p's are used in our simulations.

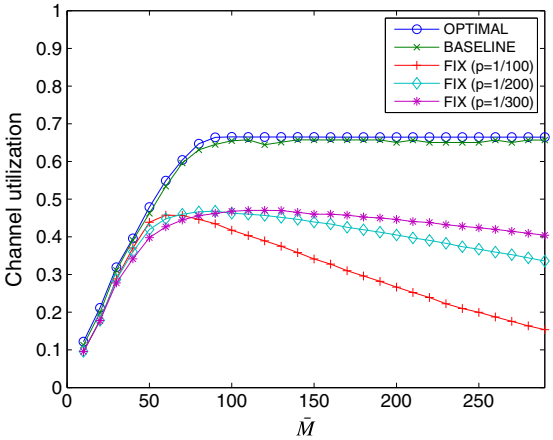


Figure 6. Comparison of channel utilization under different $\bar{M}$'s.

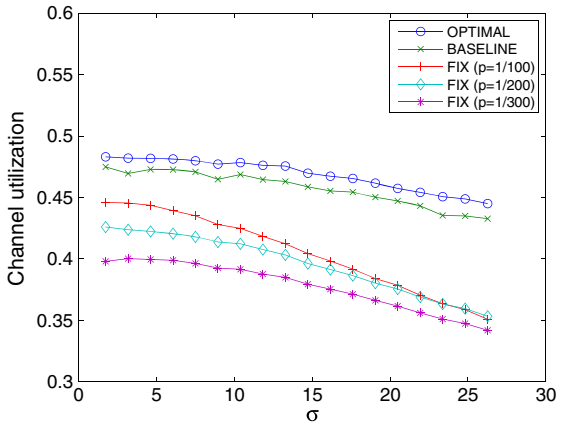


Figure 7. Comparison of channel utilization under different σ's.

The results show that there does not exist a p that provides high channel utilization for a wide range of $\bar{M}$. Even though a smaller p (i.e., 1/300) achieves better performance than larger p's when $\bar{M}$ is large, the resulting utilization is still far less than that in the proposed protocol.

4.2. The impact of σ on U

Figure 7 shows the channel utilization when $\bar{M}$ is fixed at 50 and a varies from 5 to 45. As a increases, the standard deviation σ increases, thus resulting in a more dynamic network. The figure shows that when M fluctuates more dramatically, the utilization in FIX decreases faster than our proposed protocol. The reason is that in FIX, p can only be determined on the basis of some a priori information, if such information exists. When σ increases, M fluctuates more so that the predetermined p cannot accommodate such fluctuation. In contrast, the proposed protocol tries to keep up with the fluctuation. As a result, a smaller degradation can be achieved.

5. ASYNCHRONOUS RESOURCE RESERVATION

In our baseline protocol, non-CCC channels are still wasted during the negotiation phase, even though T_n has been adjusted dynamically to minimize such waste. This is the inherent drawback in any split-phase protocol as pointed out in [9]. In what follows, we introduce two new mechanisms to further reduce the wasted channel time. The first mechanism, referred to as pair-and-go (PNG) in this paper, allows machines to utilize the non-CCC channel during the negotiation. The second mechanism, referred to as enhanced PNG, allows machines to return to the CCC at different time points (i.e., the negotiation phases may not be aligned anymore). These two mechanisms, when working together, will enable asynchronous resource reservation in a CCC-based multichannel network. The details of the proposed mechanisms are discussed in the following two subsections.

5.1. Pair-and-go

Figure 1 shows that in a split-phase protocol, all machines wait until the end of negotiation phase before starting their data transmission. For a split-phase network with a fixed T_{total}, such waiting is unnecessary and only wastes the valuable channel time. Machines that already complete their reservation should be allowed to switch to a non-CCC channel for data transmission even before the end of the negotiation phases. PNG is developed on the basis of this simple observation. By using PNG, more non-CCC channels can be utilized and a higher utilization can be achieved as illustrated in Figure 8.

Figure 9 compares the channel utilization with and without using PNG, under the same setting as in Section 4.1. The protocol with PNG is denoted as "PNG". The result shows that when the number of competing machines is large, PNG can squeeze 20% more utilization from the baseline protocol. Note that the improvement is achieved on the basis of the assumption that a reserved channel will be used up for data transmission by a single pair of machines (e.g., with bursty traffic) during the entire T_d. If not (e.g., sporadic traffic), it could be better for a machine to stay on the CCC to make more reservations so that the data transmission period is not wasted. In other words, there exists a trade-off between utilization in the negotiation phase and utilization during the data transmission phase when PNG is applied and the traffic is sporadic. We

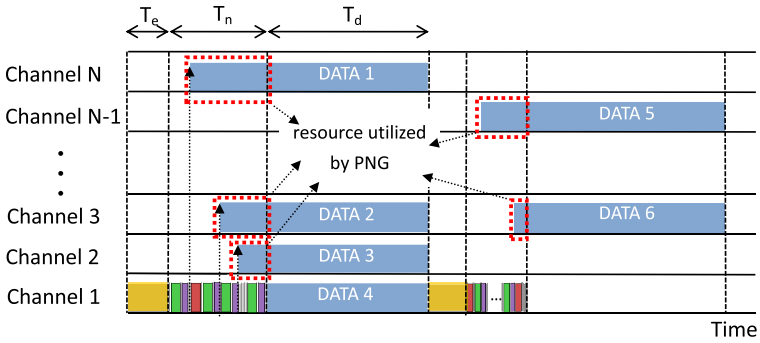


Figure 8. "Pair-and-go" in the proposed protocol.

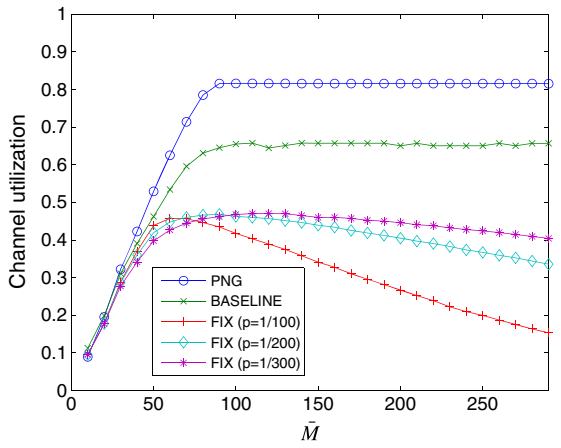


Figure 9. Comparison of channel utilization: with versus without pair-and-go.

will resolve this trade-off by using the enhanced PNG as described in the next subsection.

5.2. Enhanced pair-and-go

A quick fix for the trade-off introduced by PNG is to allow machines to return to the CCC once its data transmission is completed. By doing so, a machine can make new reservation on the CCC while others are still transmitting data in the data channel. This mechanism was used by Chen *et al.* [12] to improve the existing split-phase protocols. In their protocol, machines that have request for channel reservation run a greedy scheduling algorithm at the end of each negotiation phase. The data transmission(s) that ends first is then scheduled to the CCC as shown in Figure 10. When the data transmission(s) on the CCC ends, one of the machines on the CCC transmits a beacon to signal the start of the next negotiation phase. Therefore, data transmission and channel reservation can proceed concurrently.

One problem of asynchronous resource reservation is that the machines transmitting on the data channel (e.g., Channel 4) will lose track of new reservations in the negotiation phase. In [12], the problem is solved by including channel occupancy information in the beacon. However, as pointed out by the authors, a machine may need to wait for several negotiation phases before resynchronizing to the beacon generator. Furthermore, if the beacon generator is still on a data channel while a new negotiation phase starts, a new beacon generator needs to be elected (via contention). Both of the mechanisms will introduce additional delay and degrade the overall channel utilization.

We believe that to fundamentally solve the problems, the negotiation phase should not be synchronized at all. That is, there should be no network-wide negotiation phase. Instead, each machine maintains its own negotiation/data transmission phase based on the observed channel occupancy. This mechanism—referred to as asynchronous resource reservation—and the corresponding protocol are explained in details in the following subsections.

5.2.1. Protocol design.

Each machine using the enhanced PNG maintains a channel availability list (CAL), which records the latest time at which a channel becomes available. When a device needs to reserve a channel for data transmission, the channel that will become available first should be selected. Before accessing a selected data channel, the machine shall send an RTS frame on the common control channel first. The RTS frame is transmitted with a probability p after the CCC is determined as idle as we derived in Section 3. The network allocation vector (NAV) field of the RTS frame is set as the total time needed to transmit the buffered data frames, including all short interframe spaces and acknowledgement. If the transmission cannot be completed at the current time + the maximum channel holding time (MCHT), the NAV is adjusted to fit into the limit. The CAL is also included in an RTS to help neighboring devices keep track of the channel occupancy. In our protocol, no machine is designed as the beacon generator. Once an RTS frame is received, a machine can start its reservation on the basis of the received CAL. The RTS frame format is shown in Figure 11. Note that in our design, the first 10 bits of the NAV bits denote the reserved duration while the remaining 6 bits are used to denote the reserved channel.

Whenever an RTS frame is received, a device shall first update its local CAL as follows. For a channel whose available time is earlier than that in the received CAL, the device shall replace the available time by the value in the received CAL. Otherwise, the available time remains unchanged. If the RA field of the received RTS frame also matches the device's MAC address, a CTS frame shall be sent short

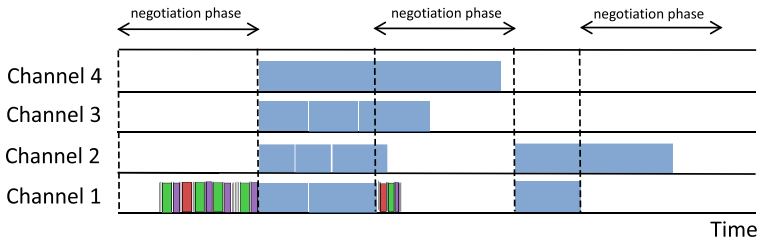


Figure 10. Concurrent resource reservation and data transmission.

	2 bytes	2 bytes	6 bytes	6 bytes	10 bits $\times (N-1)$	4 bytes
(a) RTS	Frame Control	NAV	TA	RA	CAL	FCS

	2 bytes	2 bytes	6 bytes	4 bytes
(b) CTS	Frame Control	NAV	RA	FCS

Figure 11. (a) Request-to-Send and (b) Clear-to-Send frames.

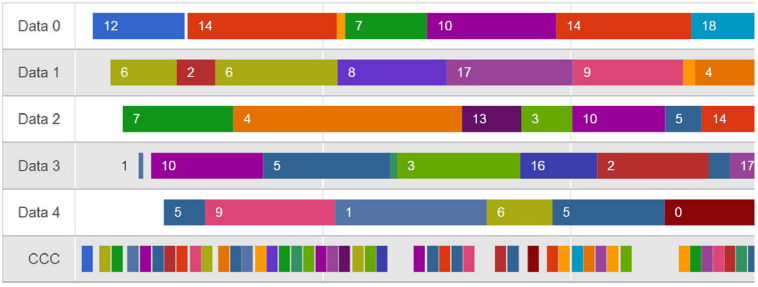


Figure 12. An example of using enhanced pair-and-go protocol: $M = 20$ and $N = 6$.

interframe space time after the reception of the RTS frame. The CTS frame format is shown in Figure 11.

Whenever a CTS frame is received and the RA field matches the receiving device's MAC address, the reservation is considered as completed. Otherwise, the receiving device simply updates its CAL on the basis of the NAV. On the basis of the aforementioned protocol, a device that just finishes its data transmission may not know the up-to-date channel occupancy because all reservations made during its data transmission were not heard. The problem can be solved partially by the CAL in the RTS frame. If a returning device receives an RTS frame on the CCC, it will learn the missing reservation. If no frame is received MCHT after the return, the device will infer that all channels are available because devices cannot make any reservation that ends MCHT after a reservation is made. An example of using the enhanced pair-and-go protocol is shown in Figure 12. The number on each colored segment indicates the ID of the device that owns the time segment for data transmission. The figure shows that the five channels are fully utilized.

5.2.2. Performance evaluation.

The performance of our enhanced PNG protocol is evaluated under different scenarios. In all of the simulations, the number of channels N is fixed at 21. The length of each RTS frame is $160 + 10 \times (N - 1) = 360$ (bits). For each machine, the inter-arrival time of data frames follows a geometric distribution with a mean equal to $\frac{1}{p_a}$, and the length is uniformly distributed between 15 bytes and 1500 bytes.

Figure 13 shows the channel utilization and average delay under different M's. The channel utilization is defined in Equation (1). The average delay is the average of the time between a data arrival and start of its transmission

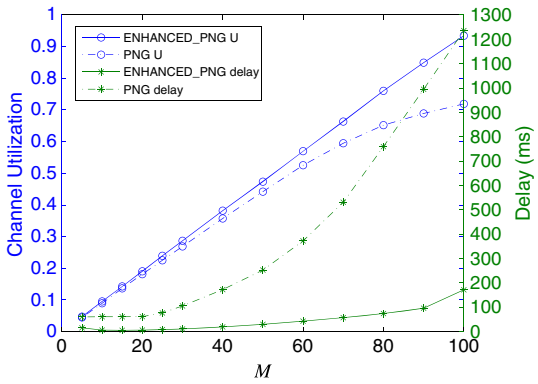


Figure 13. Comparison of channel utilization and delay under different M's.

for all machines. Here, M is varied from 5 to 100, and p_a is fixed at 0.0064. The value is selected so that the total data traffic for $M = 100$ is very close to the total channel capacity. The result shows that the utilization for $M = 100$ is 93%, which is very close to the upper bound of the channel utilization $0.95 \simeq 20/21$ given that one of the 21 channels is used as CCC. Compared with PNG, 21% more channel utilization is provided when the channels are heavily loaded, thanks to asynchronous resource reservation. An interesting observation from Figure 13 is that the average delay of enhanced PNG decreases with M first and then increases monotonically with M. The reason is that when M is small (e.g., 10), most machines switch to data channels and no one stays on CCC. When these machines return to the CCC, no RTS frame could be heard. They have to wait for MCHT before accessing the CCC, thus experiencing a longer delay than $M = 20$. When M is larger

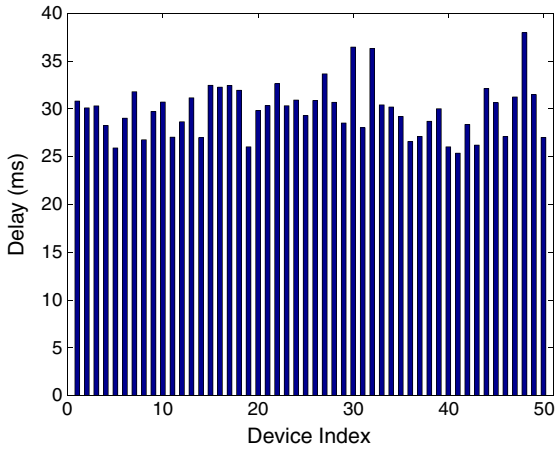


Figure 14. Average delays of individual machines for $M = 50$.

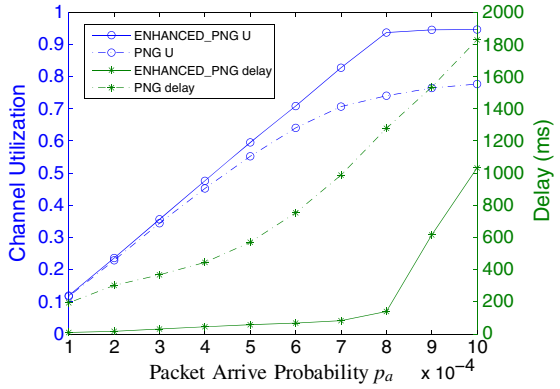


Figure 15. Comparison of channel utilization and delay under different p_a's.

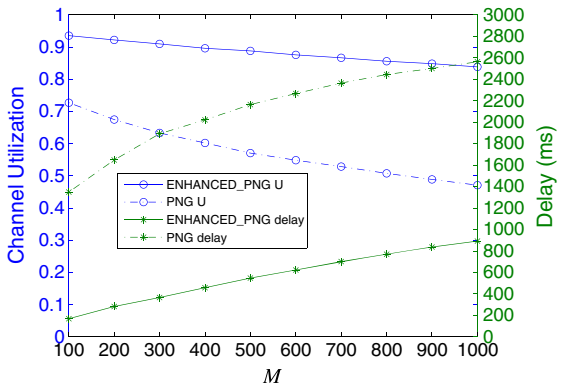


Figure 16. Channel utilization and delay in large scale networks.

than N, the average delay increases simply because of the contention on the CCC. Nevertheless, the average delay is much smaller than PNG.

In order to evaluate the fairness of the proposed protocol, the average delays of individual machines are also plotted in Figure 14. The figure shows the case for $M = 50$, $N = 20$, and $p_a = 0.0064$. The standard deviation of the delay is 2.75 ms, which is relatively small compared with the mean 29.9 ms. The fairness of our proposed protocol is achieved thanks to the equal random access probability of each machine.

Figure 15 shows the channel utilization and average delay under different p_a's with M fixed to 80. A similar trend can be observed when p_a increases from 0.0001 to 0.001. Because M is fixed to 80, the delay simply increases with p_a because of more contentions among the machines on the CCC. When p_a equals to 0.0008, the total data traffic is the same as the total channel capacity. Therefore, channel utilization saturates at 94% after p_a exceeds 0.0008. Note that when $p_a > 0.0008$, the channel resource is not enough to transmit all of the generated data. Therefore, data frames that arrive later will experience much longer delay.

Figure 16 shows the channel utilization and average delay in larger scale networks, where M is varied from 100 to 1000. For each M, p_a is set so that the generated data amount is very close to the total channel capacity. Because of the more contention from competing machines, utilization decreases slightly as the number of machines increases. The utilization of enhanced PNG when $M = 1000$ decreases only 10% compared with the utilization of $M = 100$, while the utilization of PNG decreases 35%. For every M, the average delay in enhanced PNG is also significantly smaller than that in PNG.

6. CONCLUSIONS

In this paper, we propose a distributed and adaptive CCC-based protocol for multichannel M2M networks. The optimal length of the negotiation phase and the access probability are derived to maximize the channel utilization of a baseline split-phase protocol. To further improve the channel utilization, two mechanisms referred to PNG and enhanced PNG are also developed. The simulation results show that both mechanisms provide at least 20% more channel utilization than the baseline protocol. In particular, the enhanced PNG can achieve a channel utilization of up to 93% when the traffic load is high.

ACKNOWLEDGEMENTS

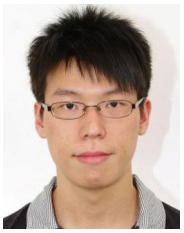
This work was also supported by the National Science Council, National Taiwan University, and Intel Corporation under grants NSC101-2911-I-002-001 and NTU102R7501.

REFERENCES

1. Shih C-F, et al. DH-MAC: a dynamic channel hopping MAC Protocol for cognitive radio networks. In *Proceedings of the IEEE International Conference on Communications*, Cape Town, South Africa, 2010; 1–5.
2. Lin Z, et al. Jump-stay based channel-hopping algorithm with guaranteed rendezvous for cognitive radio networks. In *Proceedings of the IEEE INFOCOM*, Shanghai, China, 2011; 2444–2452.
3. Shin J, et al. A channel rendezvous scheme for cognitive radio networks. *The IEEE Communications Letter* 2010; **14**(10): 954–956.
4. Hou F, et al. Asynchronous multichannel MAC design with difference-set-based hopping sequences. *IEEE Transactions on Vehicular Technology* 2011; **60**(4): 1728–1739.
5. Altamimi M, et al. Parallel link rendezvous in ad hoc cognitive radio networks. In *Proceedings of the IEEE Global Telecommunications Conference*, Miami, USA, 2010; 1–6.
6. Al-Meshhadany T, Ajib W. New multichannel MAC protocol for ad hoc networks. In *Proceedings of ICOIN*, Busan, Korea, 2008; 1–4.
7. Lo S-C, Tseng C-W. A novel multi-channel MAC protocol for wireless ad hoc networks. In *Proceedings of 65th IEEE Vehicular Technology Conference*, Dublin, Ireland, 2007; 46–50.
8. So J, Vaidya N. Multi-channel MAC for ad hoc networks: handling multi channel hidden terminals using a single transceiver. In *Proceedings of ACM MobiHoc*, Tokyo, Japan, 2004; 222–233.
9. Mo J, So H-S W, Walrand J. Comparison of multichannel MAC protocols. *IEEE Transactions on Mobile Computing* 2008; **7**: 50–65.
10. Chen W-T, Liu J-C, Huang T-K, Chang Y-C. AMMAC: an adaptive multi-channel MAC protocol for MANETs. *IEEE Transactions on Wireless Communications* 2008; **7**(11): 4541–4545.
11. Adam H, Yanmaz E, Elmenreich W, Bettstetter C. Contention-based neighborhood estimation. In *Proceedings of 71st IEEE Vehicular Technology Conference*, Ottawa, Canada, 2010; 1–5.
12. Chen J, Sheu S, Yang C. A new multichannel access protocol for IEEE 802.11 ad hoc wireless LANs. *Proceedings of IEEE PIMRC* 2003; **3**: 2291–2296.

AUTHORS' BIOGRAPHIES

Chi-Hsien Yen received his BSc degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan, in 2013. He is currently a PhD student in the Department of Computer Science at University of Illinois at Urbana-Champaign, IL, USA. His research interests include wireless communications, ubiquitous computing, and human-computer interaction.



Chen-Yu Hsu received his BSc degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan, in 2013. He will be a PhD student in the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, MA, starting in fall 2014. His current research interests include wireless communication and signal processing. Mr. Hsu received the Irving T. Ho Memorial Scholarship in 2011 and 2012, the Pan Wen-Yuan Scholarship in 2012, the Outstanding Engineering Student Scholarship from the Chinese Institute of Engineers in 2013, and the Irwin Mark Jacobs and Joan Klein Jacobs Presidential Fellowship in 2014.

Chun-Ting Chou received his BSc and MS degrees from National Taiwan University, Taipei, Taiwan, in 1995 and 1997, respectively and the PhD degree from University of Michigan, Ann Arbor, MI, USA, in 2004, all in Electrical Engineering. From 2004 to 2007, he was a senior member research staff in Philips Research North America, New York. Currently, he is an Assistant Professor in Graduate Institute of Communication Engineering, National Taiwan University. His current research interests include dynamic spectrum access (DSA), medium access control (MAC) design, wireless and mobile communications, and machine-to-machine networks.