An Adaptive Multichannel Protocol for Large-Scale Machine-to-Machine (M2M) Networks

Chen-Yu Hsu, Chi-Hsien Yen, and Chun-Ting Chou
Department of Electrical Engineering
National Taiwan University
Intel-NTU Connected Context Computing Center
{b98901172, b98901112@ntu.edu.tw, chuntingchou@cc.ee.ntu.edu.tw}

***Abstract*—With the emergence of machine-to-machine (M2M) communications, trillions of devices will soon be interconnected to support new applications and services. The success of M2M communication relies on the scalability of underlying network architectures and protocols. In this paper, an adaptive multi-channel medium access control (MAC) protocol is proposed to address the scalability issue in M2M communications. The proposed MAC protocol enables devices to (1) real-time estimate the number of competing devices and (2) adjust their operation parameters to maximize channel utilization. Our numerical results show that the proposed protocol outperforms the existing multi-channel protocols, especially when the number of competing devices is large and fluctuates with time.**

***Keywords*—M2M, Medium Access Control Protocol, Common Control Channel**

I. Introduction

M2M communication is considered as the most important evolution for the Internet after the World Wide Web (WWW). With the help of M2M communication, trillions of machines will be interconnected to support new applications and services. The key to interconnect such a huge amount of machines is the scalability of the underlying network architectures and protocols. A good protocol for M2M communication needs to scale well when the number of machines increases so that each individual machine has a fair share of resources for its data transmission.

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) are based on a multichannel architecture. However, these protocols do not fully support multichannel operation in the sense that machines do not switch among channels on a regular basis. In general, machines that need to communicate with each other, such as one WiFi access point (AP) and several associated stations, select one of the channels and compete against other "alien" machines on the same channel. As a result, the overall channel utilization is unbalanced and limited.

True multichannel operation can be supported in a centralized or distributed manner. In a centralized multichannel 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 solutions 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 multichannel network, machines negotiate with each other for channel access. Depending on how the negotiation is done, distributed multichannel networks can be further classified as channel hopping-based [1] [2] or common control channel (CCC)-based.

In hopping-based protocols, machines hop among different channels on a regular basis by following specially-designed hopping sequences. When machines that need to communicate with each other hop to the same channel, their communication can start/resume. One advantage of hopping-based protocols is that no signaling overhead is incurred given that no negotiation is needed. However, hopping-based protocols usually do not guarantee frequent "rendezvous" for communicating machines. Therefore, not only a significant portion of channel time may be wasted but also individual machines could experience long delay.

In CCC-based protocols, one of the channels is used as the control channel. On this control channel, machines negotiate with each other to reserve channels for data transmission. Since negotiation is done in advance, data transmission will be collision free. Therefore, CCC-based protocols could potentially achieve higher channel utilization than hopping-based protocols while immune to the single node failure problem in the centralized protocols. Many CCC-based protocols have been proposed for distributed multichannel networks for these two reasons. In [3] [4], a so-called dedicate control channel protocol was proposed, where each machine must equip with two transceivers. One of the transceivers is locked onto the control channel to negotiate channel reservation while the other is tuned to different channels for data transmission based on the negotiation result. By doing so, data transmission and negotiation can proceed concurrently and channels can be utilized more efficiently. The only drawbacks are that the hardware is more complicated and more power will be consumed due to the use of dual transceivers.

In order to relax hardware requirements, a split-phase multichannel protocol was proposed in [5]. In the split-phase protocol, time is divided into periodical intervals. Each interval is further divided into a negotiation phase and a data trans-

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

978-1-4673-2480-9/13/$31.00 ©2013 IEEE

mission phase. In the negotiation phase, all machines switch to the control channel to negotiate channel reservation. In the data transmission phase, machines start their data transmission in the reserved channels. Since only one transceiver is used, negotiation and data transmission cannot proceed concurrently. As a result, all channels other than the control channel will be wasted during the negotiation phase. In [6], the authors showed that channel utilization is very sensitive to the length of the negotiation phase, T_n, for split-phase protocols. However, how to determine T_n that maximizes channel utilization was not discussed. An interesting conclusion is made by the authors that the length of the data transmission phase has little impact on the selection of T_n. The conclusion is based on the assumption that all machines have no buffer space, which may not be the case even for simple machines in M2M applications. In [7], the authors proposed a split-phase protocol that adjusts T_n dynamically. The adjustment mechanism is simple but very primitive. A machine broadcasts a request for increasing or decreasing T_n after an unsuccessful negotiation or incomplete data transmission. Machines will then increase/decrease T_n by one time unit based on the majority rule. Obviously, such heuristic adjustment cannot maximize the overall channel utilization.

In this paper, we propose a distributed and adaptive CCC-based protocol for large-scale M2M networks. In order to improve channel utilization, machines using the proposed protocol estimate the number of competing machines before each negotiation phase starts. Based on the estimation result, individual 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 developed to select T_n and p such that the channel utilization can be maximized. Our numerical results show that the proposed adaptive protocol outperforms the existing CCC-based protocols, especially when the number of machines is larger and fluctuates with time.

The rest of this paper is organized as follows. In Section II, the system settings and assumptions are introduced. In Section III, the impact of T_n and p on channel utilization is analyzed. An adaptive CCC-based protocol is then proposed and the mathematical models for determining optimal T_n and p are developed. The numerical results and performance evaluation are given in Section IV. Finally, the paper is concluded in Section V.

II. System Settings and Assumptions

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 dynamic nature of M2M applications.

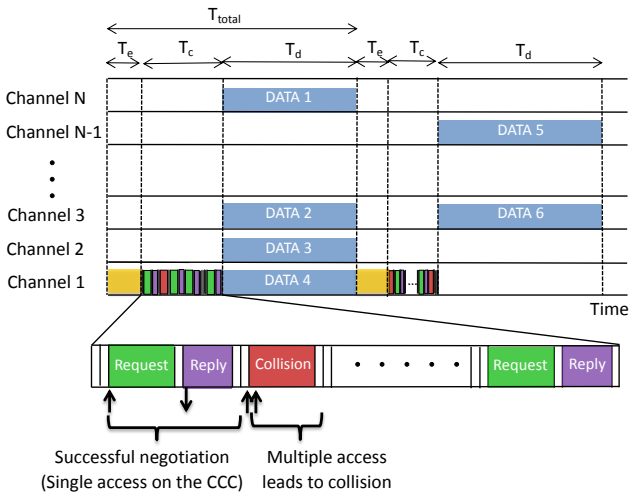

Fig. 1. Timing structure of the proposed protocol

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 successfully received a request message, the receiver waits for one inter frame space (with length of one slot), and sends back a reply message. After that, the rest of the machines wait for another inter frame space to send their request messages with a probability p. If collision happens on the CCC, all machines also wait for one inter frame 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 to $20\mu s$. These values are chosen based on 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.

III. Adaptive CCC-based Multichannel Protocol

In a CCC-based 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 since data transmission cannot proceed concurrently with negotiation under our single-transceiver assumption. Such tradeoff 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

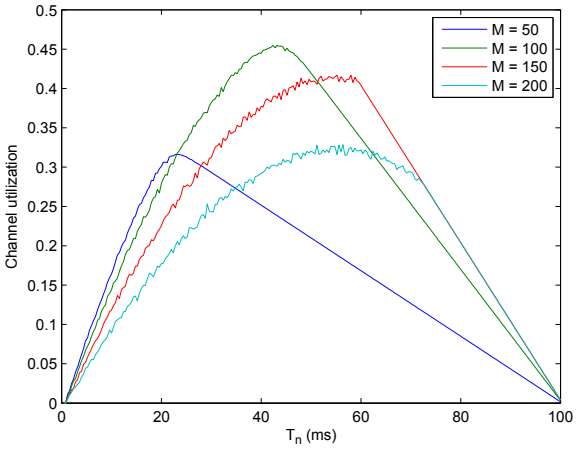

Fig. 2. Channel utilization vs. T_n: $T_{total} = 100ms$ and $p = 0.01$

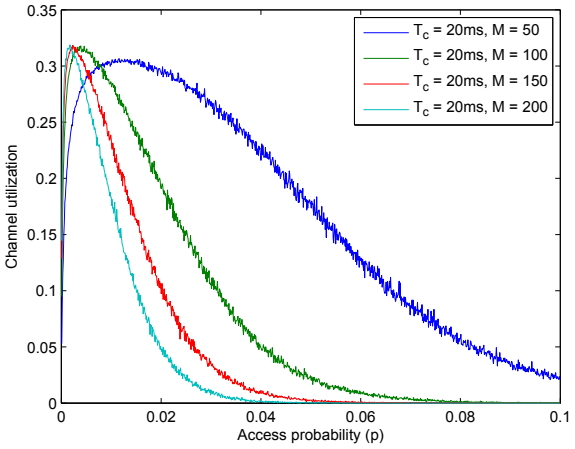

Fig. 3. Channel utilization vs. p: $T_{total} = 100ms$ and $T_n = 20ms$

utilization. Based on our findings, an algorithm to determines T_n that maximizes the channel utilization will be developed.

A. 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

$$U = \frac{T_d}{T_n + T_d} \times \frac{N_{used}}{N}, \tag{1}$$

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 (1) channel utilization varies significantly with T_n and (2) 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. Since 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 CCC-based multichannel network.

B. 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 [8], 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.

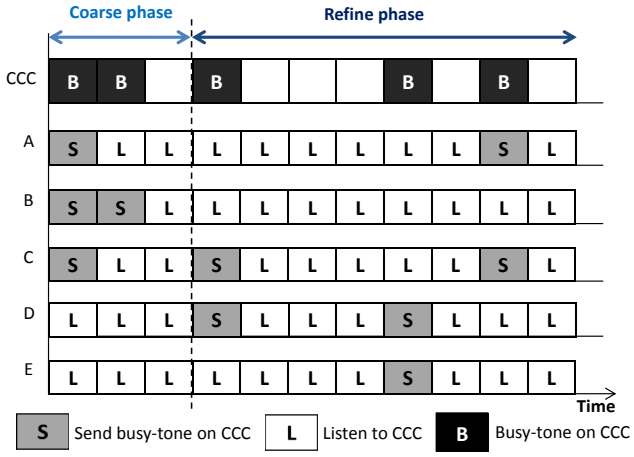

Fig. 4. An example of the proposed estimation algorithm: $M = 5$

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.

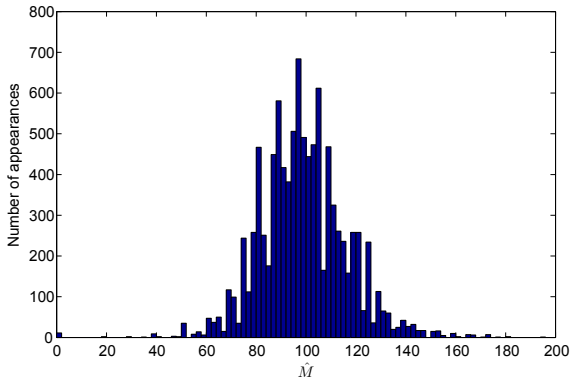

Fig. 5. Distribution of $\hat{M}$: $M = 100$ and $L_r = 100$

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 slots. Given that each machine halves the probability of sending busy tones 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. Based on the extensive numerical results, we set L_r to 100 time slots. Such a value provides a reasonable result for larger M's. At the end of the refine phase, each machine estimates the number of machines that intend to negotiate in the negotiation phase by

$$\hat{M} = \frac{log(1 - B_r/L_r)}{log(1 - p_b)}, \tag{2}$$

where B_r is 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) and p_b is the transmission probability of busy tones in each slot. An example of a refine phase with $p_b = 1/8$ and $L_r = 8$ is also shown in Figure 4. For machine A, busy tones were detected in the first and fifth slot in the refine phase, and a busy tone was sent in the seventh slot. Therefore, $B_r = 3$ in our example. By Eq.(2) $\hat{M}$ is calculated to be 3.52 machines.

Figure 5 shows the estimation results of the proposed algorithm. The figure shows the distribution of 10000 $\hat{M}'s$ for $M = 100$. 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 6 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.

C. Optimal access probability p_{opt}

As we showed in Section III-A, 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. Since 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

$$\begin{aligned} E[T_i] = {} & P_{i,1} \times \{1 + E[T_i]\} \\ & + P_{i,2}\{T_{req} + T_{rep} + 2 + E[T_{i-2}]\} \\ & + (1 - P_{i,1} - P_{i,2})\{T_{req} + 1 + E[T_i]\}. \end{aligned} \tag{3}$$

The first term in Eq (3) represents the event that none of the machine accesses the channel in the first slot. As a result, one slot time 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 $T_{req} + T_{rep} + 2$ 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 represent the event that more than one machine access the channel and collide with each others. Therefore, $T_{req} + 1$ is wasted and $E[T_i]$ is needed for these i machines to complete negotiation. The probability of this event can be obtained as $1 - P_{i,1} - P_{i,2}$.

Eq.(3) can be simplified as

$$E[T_i] = T_{rep} + 1 + E[T_{i-2}] + \frac{T_{req} - T_{req}(1 - p_i)^i + 1}{i \times p_i (1 - p_i)^{i-1}}. \tag{4}$$

In Eq.(3), p_i only appears in the last term. Therefore, the optimal p_i, $p_{i,opt}$, that minimizes $E[T_i]$ can be obtained by

$$p_{i,opt} = \arg\min_{p_i} \frac{T_{req} - T_{req}(1 - p_i)^i + 1}{i \times p_i (1 - p_i)^{i-1}}. \tag{5}$$

By simplifying Eq.(5), we can calculate $p_{i,opt}$ for a given number of remaining machines i in the network.

D. Optimal contention period, $T_{n,opt}$

It is observed from Section III-A 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., $N_{used} = \frac{90}{2} = 45$). 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 $T_{n,M,opt}$. According to Section III-C, these M machines initially must use an access probability derived in Eq.(5), $p_{M,opt}$. Once the

first pair of machines complete their negotiation, the rest of $M-2$ machines initially must use an access probability equal to $p_{M-2,opt}$. The negotiation process continues until the m^{th} pair of machines complete their negotiation (using an access probability equal to $p_{M-2m+2,opt}$).

In order to determine $T_{n,M,opt}$, 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. Based on Eq.(1), $T_{n,M,opt}$ can be calculated by maximizing the expected channel utilization $E[U]$ as

$$T_{n,M,opt} = \arg\max_{T_n} E[U] = \arg\max_{T_n} \begin{cases} \frac{T_d}{T_n+T_d} \times \frac{E[m_{T_n}]/2}{N}, & \text{if } \frac{E[m_{T_n}]}{2} < N \\ \frac{T_d}{T_n+T_d}, & \text{if } \frac{E[m_{T_n}]}{2} \geq N \end{cases}, \quad (6)$$

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 Eq.(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 III-C, the expected value of m_j, $E[m_j]$ can also be computed in a recursive way as

$$E[m_j] = P_{m_j,1,opt} \times E[m_{j-1}] + P_{m_j,2,opt} \times \{E[m_{j-(T_{req}+T_{rep}+2)}] + 2\} + P_{m_j,3,opt} \times E[m_{j-(T_{reqst}+1)}], \quad (7)$$

where $P_{m_j,1,opt}$, $P_{m_j,2,opt}$, and $P_{m_j,3,opt}$ represent the probabilities of the three events for channel access explained in Section III-C, with i replaced by m_j and p_i replaced by $p_{j,opt}$ in Eq.(5). $E[m_{T_n}]$ in Eq.(6) can be calculated using Eq.(7) with $j = T_n$. Finally, $T_{n,M,opt}$ can be obtained numerically using Eq.(6).

IV. 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}, accoridng to the protocol proposed by J. So and N. Vaidya [5] to achieve "good" performance.

For all of the simulations, the number of channels N is 40, and the length of T_{total} is 100ms (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.

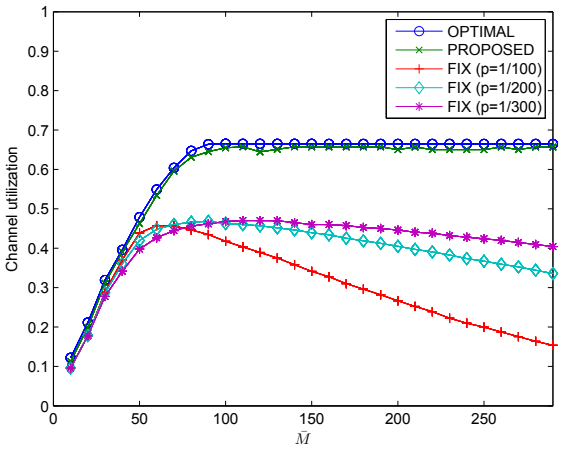

Fig. 6. Comparison of channel utilization under different $\bar{M}$'s

A. 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. 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.

B. 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 based on 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.

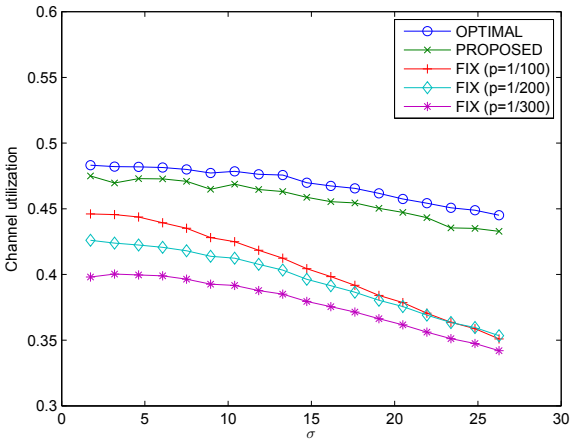

Fig. 7. Comparison of channel utilization under different σ's

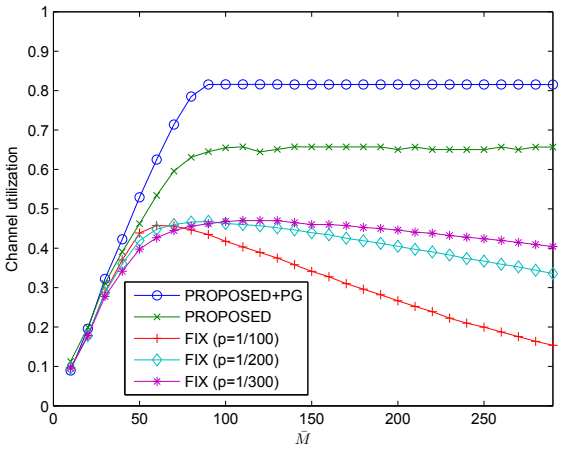

Fig. 9. Comparison of channel utilization with and without Pair-and-Go

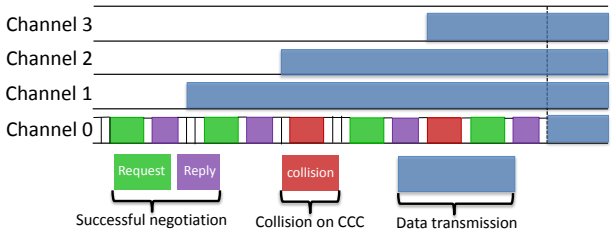

Fig. 8. "Pair-and-Go" in the proposed protocol.

C. Further Improvement of the protocol

In the proposed protocol, the channel time is still wasted in non-CCC channels during T_n, even though T_n is adjusted dynamically by the protocol to minimze such waste. This is the inherent drawback in any CCC-based protocols using a split-phase approach as pointed out in [6]. However, the problem could be further alleviated by introducing a simple rule to the proposed protocol. The rule is developed based on a simple observation. That is, two machines that successfully reserve a data channel during T_n do not need to stay in the CCC. Instead, they should switch to the reserved channel and start their transmission immediately in order to better utilize the channel. This simple rule, denoted as "Pair-and-Go" in this paper, is illustrated in Figure 8.

Figure 9 compares the channel utilization with and without using Pair-and-Go, under the same setting as Section IV-A. In the figure, the protocol with Pair-and-Go is denoted as "PROPOSED+PG". The figure shows that the saturated utilization, when $\bar{M}$ is large, increases if Pair-and-Go is adopted. The utilization in PROPOSED+PG is 20% more than the original protocol. However, it should be noted that the improvement also results from an assumption that each pair of machines uses up the reserved channel during each period. If not, an early departure from the CCC prevents each negotiated pair from knowing the reservations made later by other machines. This could be a problem if multiple non-overlapping reservations are allowed in a single channel. In the original protocol, all machines stay in the CCC so that multiple non-overlapping reservations can be made easily by adopting the channel scheduling algorithms as proposed in [9].

V. Conclusions

In this paper, an adaptive CCC-based multichannel protocol for large-scale M2M networks is proposed. The proposed protocol enables efficient estimation of the number of machines in a distributed manner. Based on the estimated number, the protocol dynamically adjusts the access probability and the length of the negotiation phase to maximize channel utilization. The numerical results show that our protocol outperforms the existing CCC-based protocols. Our protocol is a feasible solution for large-scale M2M networks, where machines may join or leave the networks dynamically.

References

[1] P. Bahl, R.Chandra, and J. Dunagan, "SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in Proc. ACM MobiCom, pp.216-230, 2004.
[2] H. So, J. Walrand, and J. Mo, "McMAC: a multi-channel MAC proposal for ad-hoc wireless networks," in Proc. IEEE WCNC, pp. 334-339, 2007.
[3] T. Al-Meshhadany and W. Ajib, "New multichannel MAC protocol for ad hoc networks," in Proc. ICOIN, pp. 1-4, Jan. 2008.
[4] S.-C. Lo and C.-W. Tseng, "A novel multi-channel MAC protocol for wireless ad hoc networks," in Proc. 65th IEEE Vehicular Technology Conference, pp. 46-50, Apr. 2007.
[5] J. So and N. Vaidya, "Multi-channel MAC for ad hoc networks: handling multi channel hidden terminals using a single transceiver," in Proc. 5th ACM Int. Symp. Mobile Ad Hoc Netw. Comput., pp. 222-233, 2004.
[6] J. Mo, H.-S. W. So, and J. Walrand, "Comparison of multichannel MAC protocols," IEEE Trans. on Mobile Comput., vol. 7, pp. 50-65 Jan. 2008
[7] W.-T. Chen, J.-C. Liu, T.-K. Huang, and Y.-C. Chang, "TAMMAC: an adaptive multi-channel MAC protocol for MANETs," IEEE Trans. on Wireless Commun., vol. 7, no. 11, pp. 4541-4545, Nov. 2008.
[8] H. Adam, E. Yanmaz, W. Elmenreich and C. Bettstetter, "Contention-based neighborhood estimation", in Proc. 71st IEEE Vehicular Technology Conference, pp. 1-5, May 2010.
[9] J. Chen, S. Sheu, and C. Yang, "A new multichannel access protocol for IEEE 802.11 ad hoc wireless LANs," in Proc. IEEE PIMRC, vol. 3, pp. 2291-2296, Sep. 2003.