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
{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.**

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 service. The key to interconnect such a huge amount of machines is the scalability of the underlying network architecture and protocol. A good protocol for M2M communication needs to scale well when the number of machines increase 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 a WiFi AP and the associated machines, 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 reponses. 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 [6] [7] 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 timely or frequent "rendezvous" to 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 [1] [2], a so-called dedicate control channel protocol was proposed, where each machine must equip with two transceivers. One of the transceivers is locked into 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 expensive and more power will be consumed due to the use of dual transceivers.

In order to relax the hardware requirement, the split-phase multichannel protocol was proposed in [3]. 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. 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 [4], 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 the optimal T_n that maximize channel utilization was not discussed. The authors only concluded that the length of data transmission phase has little impact on the optimal 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 [5], the author proposed a split-phased protocol that adjusts T_n dynamically. However, the adjustment mechanism is 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 majority rule. Obviously, such heuristic adjustment cannot maximize the overall channel utilization.

In this paper, we propose an 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 machine 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 then developed to select the optimal 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 value 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 that intend to transmit in the upcoming data transmission phase, M. In this paper, M, is assumed to be a random number given the dynamic nature of M2M applications.

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. 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. At the end of the negotiation phase, those successfully negotiated machines switch to the data channels reserved earlier and start data transmission. Throughput the 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.

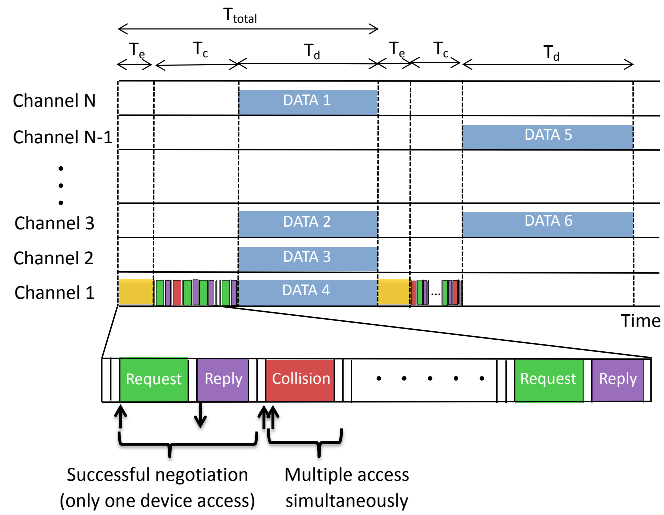

Fig. 1. Timing structure of the proposed protocol

III. Adaptive CCC-based Multichannel Protocol

In a typical 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 small, 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 large, all machines may complete negotiation. However, a large T_n implies a small 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 overall channel utilization. In what follows, we first investigate the impact of T_n on channel utilization. Based on our findings, an algorithm that determines the optimal T_n that maximizes overall channel utilization will be developed.

The negotiation process is conducted as shown in Figure 1. After successfully received the request message, the receiver will wait for one time slot of inter-frame-space, and then send back the reply message. A successful negotiation is then completed. After that, the remaining machines will wait for another inter-frame-space to send their request messages. If collision happens on the CCC, all machines will also wait for one inter-frame-space to resend their request messages. In Figure 1, the one more slot between the request message the collision is because no machine send request message in that slot.

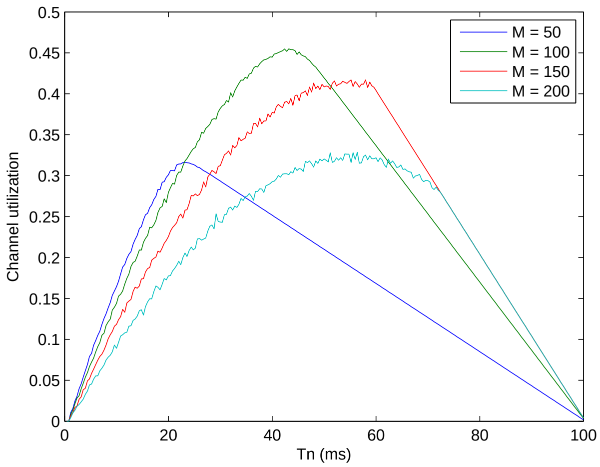

Fig. 2. Channel utilization vs. T_n: $T_{total} = 100ms$ and $p = 0.01$

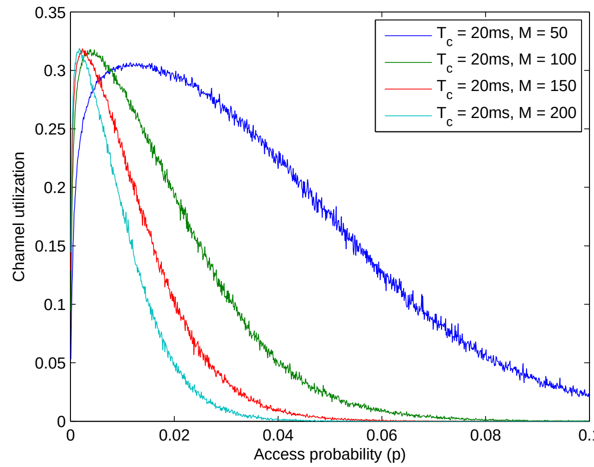

Fig. 3. Channel utilization vs. p: $T_{total} = 100ms$ and $T_n = 20ms$

A. Impact of T_n on channel utilization

We consider a large-scale 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 channels used 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 channel utilization varies significantly with T_n and there exists an optimal T_n that maximizes the channel utilization. Take $M = 100$ in Figure 2 as an example. Choosing $T_n = 20$ms, which is close to the optimal T_n for $M = 50$, will degrade the utilization 37% when compared to the optimal value for $M = 100$. The optimal T_n depends on M. However, when M changes frequently in large-scale networks, optimal T_n cannot be determined 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, the number of machines that complete negotiation. Therefore, p will also determine 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 when M varies frequently. The study illustrates that dynamic adaptation of p and T_n to the change in M will be the key to the efficiency of a CCC-based multichannel network.

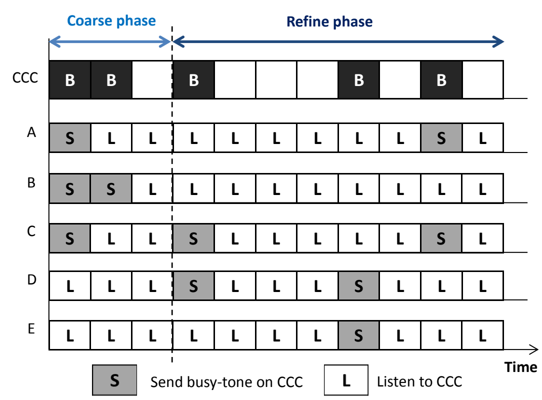

Fig. 4. An example of the proposed estimation algorithm: $M = 5$

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 [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, all of the machines send 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 will send 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

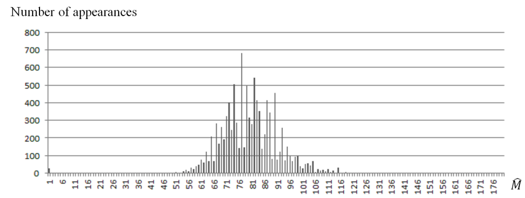

Fig. 5. The distribution of $\hat{M}$: $M = 80$ and $L_r = 100$

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 buys tones in the third time slot. In this case, the coarse phase is completed in the third slots. Given that each machines halves the probability of sending busy tones exponentially every time slot, the average length of 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 estimate the number of machines that intend to negotiation 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. The 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, so $B_r = 3$ in this case. 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 estimated $\hat{M}$. The results show that the average of $\hat{M}$ is equal to M and the standard deviation is 11.6. The average total time for each estimation is 107 time slots, which are only about 6 request messages long. Consider there are 80 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 when i machines use for negotiation. Define T_i as the time slots needed for i machine to complete all negotiations. The expected value of T_i can be computed in a recursive way as

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

The first term in Eq (3) represents the event that none of the machine access 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 only one machine access the channel and thus, completes negotiation with its target devices. A total of $T_{req} + T_{rep} + 2$ is needed for the two machines to complete the negotiation and $E[T_{i-2}]$ is still 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 the same number of 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

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

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

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

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 T_n, only 90 machines complete negotiation (i.e., $N_{used} = 45$). 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), the $T_{n,M,opt}$ can be calculated by maximizing the expected channel utilization $E[U]$ as

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

where $E[m_{T_n}]$ represent 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 will 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

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

where $P_{m_j,1,opt}$, $P_{m_j,2,opt}$, and $P_{m_j,3,opt}$ represents 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 $p_{j,opt}$ defined 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 then be obtained numerically using Eq.(6).

IV. Numerical Analysis and Evaluation

In this section, we compare the channel utilization of the proposed adaptive CCC-based multichannel protocol, ADMAC, with other wireless multichannel protocols, OPTIMAL and FIX ($p = 1/100$, $p = 1/200$, and $p = 1/300$) protocols. OPTIMAL is an imaginary ideal protocol that knows M without estimation, and apply same procedures of our proposed protocol in T_n and T_d. OPTIMAL is used to compare with the performance of proposed protocol and provide an optimal bound of utilization. As for FIX protocols, they do not estimate M, and the parameters such as T_n and p are fixed. According to the protocol proposed by J. So and N. Vaidya [3], the length of T_n is 20% of T_{total} in FIX protocols.

In the scenarios of following experiments, the number of channel N is 40, and the length of T_{total} is 100ms, i.e., 5000 time slots. The number of machines M is a uniform distributed random variable and is different in each T_{total}. $\bar{M}$ is the mean of M and $\mathcal{M}$ is the sample space of M, i.e., $M \in \mathcal{M} = [\bar{M} - a, \bar{M} + a]$, where the variance $Var[M] = ((2a+1)^2 - 1)/12$. The following two experiments varies $\bar{M}$ and $Var[M]$ to observe their effects on the protocol utilization.

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

Figure 6 compares the utilization with different values of $\bar{M}$, varies from 10 to 300, while a is fixed to 10. As the result shows, both the channel utilizations of OPTIMAL and ADMAC protocols increase as $\bar{M}$ gets larger until the all data channels are fully reserved, i.e., $\bar{M} = 80$ $(= N \times 2)$. The utilization increases because all of the devices could complete negotiation and transmit data in the channel. When $\bar{M} > 80$, the protocols will allow only 80 devices to complete negotiation in T_n. In addition, the efficiency of negotiation is not affected by $\bar{M}$. Therefore, the utilization is not changed when $\bar{M} > 80$. The small gap between ADMAC and OPTIMAL also proves that T_e incurs little overhead and could effectively improve the utilization. In the FIX protocols, the number of devices is not estimated, so that p has to be fixed. As the result shows, the three FIX protocols outperforms each other for different $\bar{M}$'s, For example, when $\bar{M} = 50$, FIX protocol with $p = 1/100$ has the highest utilization among the FIX protocols. However, when $\bar{M} = 80$, FIX protocol with $p = 1/200$ performs best, and when $\bar{M} \geq 100$, FIX protocol with $p = 1/300$ has the highest utilization. When M fluctuates with time, without an accurate estimation of M, the p of FIX protocols could not be properly determined to maximize the channel utilization.

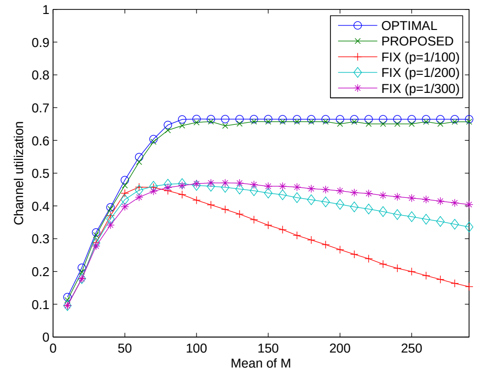

Fig. 6. The comparison of channel utilization under different $\bar{M}$'s in different protocols

B. The impact of $Var[M]$ on U

Figure 7 shows the utilization of the scenarios where $\bar{M}$ is fixed to 50 and a varies from 5 to 45. As a gets larger, the standard deviation $\sigma = \sqrt{Var[M]}$ increases, thus representing a more dynamic network. The figure shows that when M fluctuates more dramatically, the utilization of FIX protocols will decrease more than our proposed protocol. The reason is that in FIX protocols, p is determined based on a fix value of M. Therefore, When σ gets larger, M will deviate from the fix value more frequently, and the chance of p being unsuitable to M gets larger. In most of the scenarios of real applications, such as the M2M traffic system, the number of devices could fluctuate from time to time. As a result, the proposed protocol will performs better in real world.

C. Further Improvement of the protocol

In our protocol, though T_n is adjusted dynamically for optimized utilization, the channel resources wasted in other data channels during T_n still become the bottleneck for our

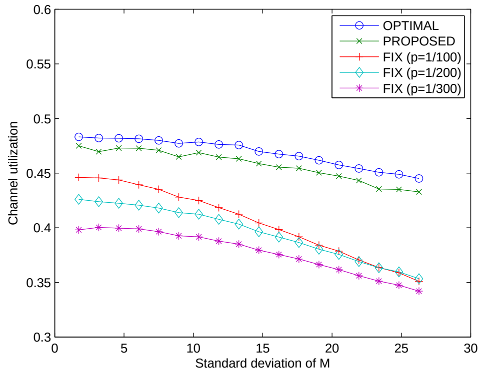

Fig. 7. The comparison of channel utilization under different $Var[M]$'s in different protocols

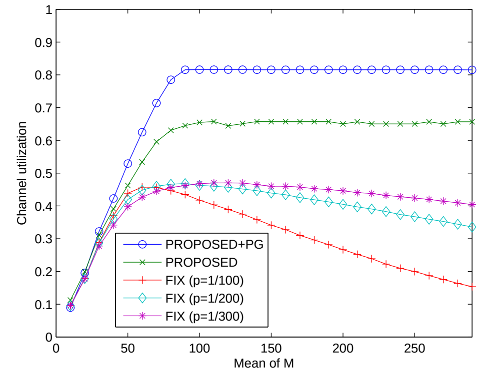

Fig. 9. Comparison of Pair-and-Go and AD-MAC.

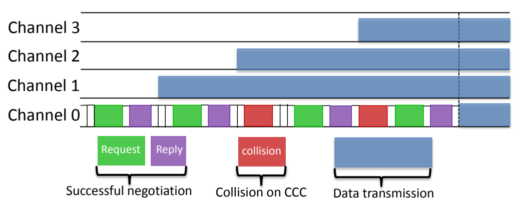

Fig. 8. "Pair-and-Go" in the proposed protocol.

split-phased CCC-based protocol. In fact, this has long been considered a major drawback for split-phased CCC-based protocols as [4] had pointed out. However, the problem could be eliminated by introducing a new feature in our proposed protocol. In fact, machines who successfully reserve the data channel earlier during T_n do not need to record the channel reservation status after the success. In order to better utilize data channels during T_n, we introduced the "Pair-and-Go" concept: once the negotiation between two machines is done, the machine pair can switch to the reserved data channel and start transmitting data immediately (as shown in Figure 8).

Adopting the same settings as in Section IV, Figure 9 combines the numerical results of "Pair-and-Go" protocol (PROPOSED+PG) with previous results in Figure 6. The utilization improvement of PROPOSED+PG over PROPOSED increases as the number of machines M in the network increases. When M is larger than twice of N (N = 40), the utilization improvement of PROPOSED+PG over PROPOSED will be saturated at about 20%.

However, a trade-off exists between using Pair-and-Go mechanism or not in our proposed protocol. Without Pair-and-Go, our proposed protocol could be easily adapted to use channel scheduling as described in [12]. In negotiation phase, the devices claim how long they need to transmit data for each receivers. Then, a channel scheduling algorithm is adopted to assign multiple devices to one data channel and fully utilize the data channels during negotiation phase. However, if Pair-and-Go is adopted in our proposed protocol, the devices who complete negotiation has to switch to the data channel immediately. As a result, the devices could not negotiate with other devices in the negotiation phase again and thus decreases the efficiency of channel scheduling.

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 adjust the access probability and the length of the negotiation phase to maximizes channel utilization. The numerical results show that our protocol outperform 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] T. Al-Meshhadany and W. Ajib, "New multichannel MAC protocol for ad hoc networks," in Proc. ICOIN, Jan. 2008, pp. 1-4.

[2] S.-C. Lo and C.-W. Tseng, A novel multi-channel MAC protocol for wireless ad hoc networks, in Proc. IEEE 65th VTC-Spring, Apr. 2007, pp. 46-50.

[3] 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., New York, 2004, pp. 222-233.

[4] Jeonghoon Mo; So, H.-S.W.; Walrand, J.; Comparison of Multichannel MAC Protocols, Mobile Computing, IEEE Transactions on, Jan. 2008

[5] W.-T. Chen, J.-C. Liu, T.-K. Huang, and Y.-C. Chang, TAMMAC: An adaptive multi-channel MAC protocol for MANETs, IEEE Trans. Wireless Commun., vol. 7, no. 11, pp. 4541-4545, Nov. 2008.

[6] P. Bahl, R.Chandra, and J. Dunagan, "SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks," in MobiCom, 2004.

[7] H. So, J. Walrand, and J. Mo, "McMAC: A multi-channel MAC proposal for ad-hoc wireless networks," in Proc. IEEE WCNC, 2007, pp. 334-339.

[8] Z. Naor, and H. Levy, "A Centralized Dynamic Access Probability protocol for next generation wireless networks", INFOCOM 2001, vol. 2, pp.767-775

[9] B. F. Lo, "A survey of common control channel design in cognitive radio networks", Physical Communication 2011, pp.26-39

[10] C. Liu, Z. Wang, and H. Song, "Design on Common Control Channel of MAC protocol of Cognitive Radio Networks" 2010 International Conference on Electrical and Control Engineering, pp.3621-3624

[11] H. Adam, E. Yanmaz, W. Elmenreich and C. Bettstetter, "Contention-based neighborhood estimation", Vehicular Technology Conference, IEEE 71st, 2010, pp. 1-5.

[12] J. Chen, S. Sheu, and C. Yang, A new multichannel access protocol for IEEE 802.11 ad hoc wireless LANs, in Proc. 14th IEEE PIMRC, Sep. 2003, vol. 3, pp. 2291-2296.