This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2010 proceedings.

EMAC: An Asynchronous Routing-Enhanced MAC Protocol in Multi-hop Wireless Networks

Shu Du* Yanjun Sun* David B. Johnson†
*Systems and Applications R&D Center, Texas Instruments, Dallas, TX, USA
†Department of Computer Science, Rice University, Houston, TX, USA

***Abstract* — Traditional contention-based random-access wireless MAC protocols such as IEEE 802.11 DCF are designed for single-hop wireless networks and do not perform well in multi-hop scenarios due to inefficiency in their medium reservation mechanisms. In this paper, we introduce a new MAC protocol, called EMAC, which improves the efficiency of wireless medium reservation for general asynchronous multi-hop wireless networks. By exploiting limited routing information at the MAC layer, EMAC enables multiple asynchronous stations along a delivery path to cooperate in their random medium access. In particular, a control frame can travel across a multi-hop path composed of *asynchronous* nodes, and make *synchronized* medium reservations for an upcoming data frame transmission. This distributed cooperation at the MAC layer can greatly improve the medium reservation efficiency by reducing intra-flow contentions. Simulation results show that EMAC can improve end-to-end throughput compared to IEEE 802.11 DCF in chain scenarios by up to 85% and can avoid throughput starvation in cross scenarios.**

I. Introduction

Many types of multi-hop wireless networks are currently being studied, such as mobile ad hoc networks (MANET), wireless sensor networks (WSN), and wireless mesh networks. Despite the significant research on these networks, little work has focused on medium access control (MAC) protocols specifically for multi-hop wireless relaying, a basic foundation of all the multi-hop wireless networks. Research in multi-hop wireless networking has typically used traditional *single-hop* MAC protocols such as IEEE 802.11 DCF [1], with multi-hop delivery being performed in a store-and-forward manner. However, using single-hop MAC protocols in multi-hop environments has some limitations. For example, if IEEE 802.11 DCF is used in a multi-hop network, a downstream node must compete with its own upstream node along the same forwarding path for transmission opportunities, even when both nodes are serving the same traffic flow. This intra-flow contention can significantly degrade network performance [2], [3].

We believe that CSMA/CA medium access efficiency can be improved if multiple hops along the path coordinate some rather than just using "random" access. In this paper, we describe the design and evaluation of EMAC (Express MAC), a new, efficient multi-hop CSMA/CA MAC protocol for general *asynchronous* multi-hop wireless networks; we extend our prior work on RMAC [4], which was originally designed only for *synchronous* sensor networks with light traffic loads.

This work was supported in part by the U.S. National Science Foundation under grants CNS-0520280, CNS-0435425, and CNS-0325971.

EMAC is quite different from RMAC in a number of aspects. First, RMAC works only when all nodes have synchronized clocks, which significantly constrains its usage; EMAC, instead, is designed for general multi-hop wireless networks and does not assume clock synchronization. Second, RMAC works best for light traffic loads typical for WSNs, whereas EMAC can support higher, more varied traffic loads. Finally, RMAC is a duty-cycling MAC protocol, designed for energy-constrained applications, whereas EMAC is a general CSMA/CA MAC protocol designed for many types of general applications. such as wireless mesh networks or wireless backbone networks.

EMAC introduces *synchronized* intra-flow coordination across multiple *asynchronous* hops while still using CSMA/CA to randomly access the wireless medium and to alleviate hidden terminal problems. Our simulation results show that EMAC can significantly improve end-to-end network throughput by reducing intra-flow contention.

II. EMAC Protocol Design

Figure 1 shows an overview of the operation of EMAC. Node S here has a data packet to send to some node several hops away. Node S transmits a PION (Pioneer) control frame to the downstream node A, which relays the PION to B. The PION may be relayed again by B if B is not the final destination of the data packet. Similar to an RTS in IEEE 802.11, a PION is used for requesting communication to a downstream node; simultaneously, forwarding the PION is used for confirming the PION receipt from the upstream node, like a CTS in IEEE 802.11. If S receives the forwarded PION from A, S will then send the data frame to A after a scheduled delay that allows the PION to be further forwarded downstream without interfering with the data transmission. In order to handle asynchronous clocks at distributed nodes and to maximize network throughput, nodes in EMAC calculate and maintains scheduling information such as $T_{delay}, t_0, ..., t_3$ (shown in the figure and described more fully below) in a distributed manner using only their local, unsynchronized clocks.

A. Pioneer (PION) Frame Transmission

A PION frame in EMAC must contain the necessary information, such as the Final Destination Address and Hop Count, to allow relaying nodes to forward the PION and reserve the medium for the upcoming data frame over multiple hops across the routing path. Different from RMAC, EMAC's PION frame also contains a field called *Data Delay Factor*,

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

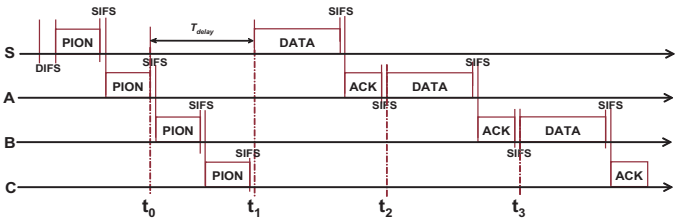

Fig. 1. Overview of EMAC operation. $T_{delay}, t_0, ..., t_3$ are calculated at distributed nodes in order to efficiently handle high traffic loads in an asynchronous multi-hop wireless network without synchronized clocks.

which controls the delay between reception of a PION and transmission of the corresponding data frame at the originating node. The details of the usage of the Data Delay Factor are described in Section II-B.

When the network layer passes a packet to the MAC layer and initiates a new EMAC transaction, EMAC builds a PION frame and sends it to the next hop node. When a relaying node receives the PION, it uses the Final Destination Address field to look up (e.g., using information from the node's routing table) the MAC address of the further next-hop node along the existing route to the packet's destination. When the PION is transmitted by a relaying node, the previous upstream node also receives the PION and interprets it as an acknowledgment.

The PION relaying process continues until the final destination is reached, until a dropped PION interrupts the relaying process, or until the current requested transmission schedules conflict with any existing schedule (details in Section II-C).

For example, in Figure 1, a PION sent by A serves both as a confirmation to the PION from its upstream node S and as a request for medium access to its downstream node B. The originating node S will retry a PION transmission if a confirmation PION does not arrive in time, as in IEEE 802.11. An intermediate relaying node, however, does not retry PION transmission if it does not receive the confirmation PION from downstream.

B. Data Transmission

Without synchronized clocks, two new challenging problems, not present in RMAC, must be solved to use the PION mechanism in EMAC. First, a relaying node must determine the time at which the data frame transmission starts at the hop-0 node along the path. Only after that, a relaying node can further predict when the data will arrive based on the Data Frame Duration and the Hop Count carried in the corresponding PION. Second, the schedule for data frame transmissions must ensure that data frames do not to collide with the ongoing PION transmissions.

In Figure 1, node S must wait for an extra time period T_{delay} before starting to transmit the data frame to A. This delay is necessary, as if S transmits the data frame immediately after receiving the PION from A, the data transmission may collide at A with another PION transmission from a downstream node, such as nodes B or C.

The hop-0 node (PION frame) determines the value of T_{delay} mainly based on the path length. If the path length is 1, then

$$T_{delay} = \text{SIFS} \ . \tag{1}$$

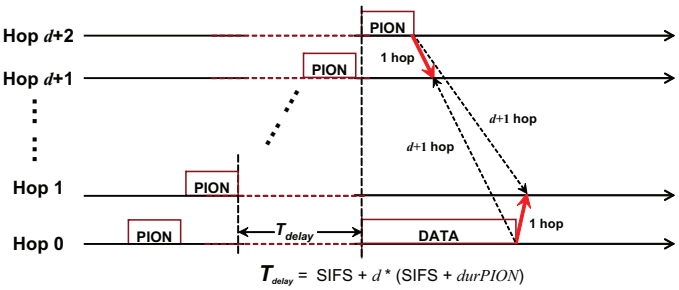

Fig. 2. Setting T_{delay} when the path length is greater than 1, using Equation 2. Interfering transmissions are $d+1$ hops away.

If the path length is greater than 1, then

$$T_{delay} = \text{SIFS} + d \cdot (T_{pion} + \text{SIFS}) \ . \tag{2}$$

In the equations, d is the Data Delay Factor carried in the PION and should be set by the hop-0 node based on network characteristics. The hop-0 node uses d to tell all the nodes along the path about the selection of T_{delay}. The value T_{pion} in the equation is the time required to transmit a single PION.

If the path length is 1, the data frame transmission will never collide with a downstream PION transmission since there is no downstream hop. In this case, the Data Delay Factor is set to zero in the PION by the hop-0 node, informing all one-hop neighbors of these two nodes that the node originating the PION has set T_{delay} according to Equation 1.

If the path length is greater than 1, as shown in Figure 2, the node originating the PION determines T_{delay} according to Equation 2, ensuring that when the hop-1 node starts receiving the data frame from the hop-0 node, any downstream node still relaying the PION is at least $d+1$ hops away, where d is the value of the Data Delay Factor in the PION.

An exception in using Equation 2 occurs when the Data Frame Duration T_{data} (the time to transmit the data frame) is smaller than T_{pion}. In this case, an extra adjustment of $T_{pion} - T_{data}$ should be added to the T_{delay} calculated by Equation 2. Considering the PION is just slightly larger than an ACK frame, the PION frame relaying process will never be caught up to by the data frame relaying process, no matter how small the data frame payload is, because the data relaying process over one hop actually consists of the transmission of the data, a SIFS time, and the transmission of an ACK.

C. Scheduling Using Local Clocks

Because there is a gap between the time when a node relays a PION and the time the corresponding data frame actually arrives, it is possible that another PION from a different flow may arrive during the idle time within that gap. For example, in Figure 1, another node Q may forward a new PION to C for a different flow shortly after t_1, This new PION and its corresponding data would very likely interfere with the data from node S if node C does not handle the new PION cautiously. Therefore, other than using NAVs as in IEEE 802.11, EMAC also introduces the mechanism of Transmission Commitment (TC) at a relaying node to avoid potential scheduling conflicts for correct and efficient medium reservation. A TC is a time segment during which the relaying node will be busy for either receiving or transmitting. A TC

is calculated based on a nodes' *local* clock, which makes it suitable for a network without synchronized clocks. RMAC does not have such mechanism, as it is only optimized for light traffic loads, under which the scheduling conflicts are unlikely to occur, but EMAC is intended for higher or more varied traffic loads.

Also, the most challenging problem in asynchronous networks is that all timings in TCs and NAVs have to be based on a node's *local* clock.

If a node receives a PION with Hop Count h, it can estimate when the data frame will be sent at the hop-0 node based on h, T_{pion}, and T_{delay}. The node can first calculate the time when the hop-0 node receives the confirmation PION from the hop-1 node. If we define the time as t_0, as shown in Figure 1, then

$$t_0 = NOW - (h-1) \cdot (T_{pion} + \text{SIFS}), \quad (3)$$

in which *NOW* is the current local time when the PION is received. T_{pion} + SIFS is the total time needed to relay the PION frame across one hop. If the data frame starting time at the hop-0 node is defined as t_1, as shown in Figure 1, then

$$t_1 = t_0 + T_{delay}, \quad (4)$$

in which T_{delay} is calculated based on d in the PION and the corresponding Equation 1 or 2. Finally, if the data frame starting time at the xth node (hop-$(x-1)$ node) along the path is defined as t_x, then

$$t_x = t_1 + (x-1) \cdot (T_{data} + \text{SIFS} + T_{ack} + \text{SIFS}), x \geq 1, \quad (5)$$

in which T_{data} + SIFS + T_{ack} + SIFS is the total time needed to deliver the data frame across a single hop. For example, in Figure 1, node B is the third node (also the hop-2 node, relative to the originating node S) along the path. Therefore, B starts data frame transmitting at t_3.

By using Equation 5 above, every PION receiver can predict in local time when the scheduled data frame will begin to arrive at any given node along the path. Correspondingly, they can also set up their TCs (if they are relayers) and NAVs (if they are neighbors of relayers) based on their local clocks.

The TC and NAV schedules of a node must be considered when making decisions on PION relaying: If the requested TC for a newly arrived PION conflicts with the current TCs or NAVs of this node, depending on the conflict situation, the node should stop the PION relaying by dropping the PION silently or by sending a CTS-only PION (to break the PION relaying).

D. Exception Handling

We emphasize that although EMAC introduces medium reservation across multiple hops, it is still a pure CSMA/CA-based *random access* protocol. Frames may experience collisions from hidden nodes, which further causes invalid medium reservations. Similar with IEEE 802.11, EMAC works correctly even when packet drops occur; MAC-level retries will recover the error and deliver the data to the final destinations. If a data frame does not get delivered to the final destination in a single EMAC transaction, the data frame's last receiver will pass the data frame from the MAC layer to the routing layer, which will later issue a new EMAC transaction. EMAC is resilient to errors: when the channel is bad or collisions are high, EMAC will gracefully degrade to shorter relays (fewer number of hops in a single transaction) or even 1-hop relays, behaving the same as IEEE 802.11. On the other hand, whenever there are opportunities, EMAC will exploit relaying a data frame further in a single MAC transaction, consequently improving network performance.

It is possible that an invalid medium reservation may propagate to further downstream due to a missed PION acknowledgment. To address this, EMAC also has several schedule cancellation policies to prevent obsolete schedules (TCs and NAVs) from reducing network performance. In general, if a data frame does not arrive at the expected time, the node cancels all the relevant schedules for the data. Also, if a new PION is received from an upstream node requesting a new schedule that conflicts with the schedules that were setup for the same upstream node, the relevant old schedules are canceled. Details about the EMAC schedule cancellation policies are omitted here due to space limitations.

TABLE I
NETWORKING PARAMETERS

Basic Rate	1 Mbps	Data Rate	2 Mbps
Tx Range	250 m	Carrier Sensing Range	250m
Capture Threshold	10 db	Contention window (CW)	31∼1023
SIFS	10 us	DIFS	50 us
Slot time	10 us	Routing processing delay	25 us

III. EMAC EVALUATION

This section gives a simulation-based evaluation of EMAC using version 2.29 of the *ns-2* simulator. The EMAC simulation module is implemented based on the IEEE 802.11 module that is distributed together with *ns-2*. In the simulations, a single omni-directional antenna is used at each node, with radio propagation modeled by the common combined Free Space and Two-Ray Ground reflection model. Our main purpose is to evaluate the EMAC's PION multi-hop reservation mechanism against RTS/CTS's single-hop reservation mechanism. Similar to IEEE 802.11 in which a packet-size threshold is used to control whether to use RTS/CTS, EMAC can be easily extended to have a similar threshold so that small packets do not use PION control frames. Therefore, we compare EMAC against both IEEE 802.11 DCF when RTS/CTS is always used and when it is never used, essentially representing the case when this threshold in 802.11 is set to the minimum packet size and the case in which it is set to the maximum packet size, respectively, bounding the performance of IEEE 802.11 DCF relative to that of EMAC.

A. Overview of the Simulation Setup

Table I shows the key parameters used in our simulations. We set the EMAC Data Delay Factor (d) to 2 here, as we believe in most of the cases, simultaneous wireless transmission from 3 hops away will not interfere with a local transmission and reception; indeed, as indicated in [5], if using the two-ray ground reflection model to analyze, a capture threshold of 10dB is sufficient to accept any d value greater than or equal to 1, assuming hop distances are the same. Detailed discussion and evaluation of the other values of d is beyond the scope of

TABLE II
TRANSMISSION DURATION PARAMETERS

Frame Type	Frame Size (bytes)	Tx Latency (us)
RTS	20	352
CTS/ACK	14	304
PION	28	416

Fig. 3. Chain scenario

this paper. In our simulations, traffic loads are generated by constant bit rate (CBR) UDP flows. The UDP packet size is fixed during one simulation. When a data frame is transmitted, the frame includes a 28-byte MAC header (in both EMAC and IEEE 802.11) and a 20-byte IP header, in addition to the UDP payload. The frame sizes and the transmission durations of different types of control frames are shown in Table II. We set the PION frame size 8 bytes larger than a RTS frame (28 bytes, total) to carry the extra information mentioned in Section II-A. The transmission duration in this table is based on the 1 Mbps basic transmission rate and 24-byte PLCP header size.

We use two basic types of scenarios in our simulations, chain scenarios and cross scenarios, as illustrated in Figure 3 and Figure 4, respectively. In a chain scenario, nodes are deployed along a straight line. Neighboring nodes are 200 meters apart, which is just within the range of a single wireless transmission hop (250 meters). One CBR flow sends packets from one end of the chain to the other end. For a cross scenario, it is composed of two chains deployed across each other. One node is shared by both chains at their midpoints. Two CBR flows send packets independently, one from the end of each chain to the opposite end of the same chain. Packets of the two flows are generated at the same time and at the same rate to create inter-flow contention in the crossing area. Also, we used an example of realistic scenario in our simulations, shown in Figure 5. This scenario has 200 nodes randomly distributed in a 2000 meters by 2000 meters square area.

We focus our evaluation of EMAC against IEEE 802.11 in this paper under backlogged UDP traffic loads instead of less challenging light traffic loads. All backlogged CBR flows generate UDP packets at 1000 packets per second, which is able to saturate the network with just one flow. We used four different packet sizes: 50, 500, 1000, and 1500 bytes. However, limited by space in this paper, we mainly focused here on the results for 1500 bytes. More results from the other packet sizes and the results from the non-backlogged traffic can be found in [5]. Each simulation runs for 10 seconds of simulation time. For each network configuration, the average over 4 runs with random seeds is reported; the standard deviation is shown as the error bars.

B. Evaluation in Chain Scenarios

We evaluated EMAC in the chain scenarios, with the length of the chains varied from 1 hop to 14 hops.

Figure 6 shows the change in average end-to-end throughput in the chain scenarios with different path lengths when the packet size is 1500 bytes. All three protocols (EMAC, IEEE 802.11 with RTS, and IEEE 802.11 without RTS) decrease significantly in throughput as the path length increases when the path length is smaller than 4 hops, which is because the medium is shared by more links as path length increases.

When the path length is 4 hops and more, EMAC shows the best throughput among the three protocols. Particularly, EMAC shows significant throughput improvement over IEEE 802.11 with RTS, suggesting the advantages of PION mechanism over RTS/CTS. When path length is 14 hops, for example, EMAC (with a throughput of 427.5 kbps) delivers 24.5% more packets than does 802.11 without RTS (with a throughput of 343.2 kbps) and 85.7% more packets than does 802.11 with RTS (with a throughput of 230.1 kbps). In these scenarios, intra-flow contention is the major factor determining the network performance. EMAC's multi-hop delivery capability therefore can be very useful here, since a packet from a node can be delivered multiple hops away before the node begins to send the next packet, significantly reducing the intra-flow contention between one node and its downstream nodes.

To show EMAC's performance with different packet sizes, Figure 7 shows the average end-to-end throughput in the 8-hop chain scenario with four different packet sizes. EMAC generally shows its performance advantage over IEEE 802.11 increases as the packet sizes becomes larger, except that IEEE 802.11 without RTS shows the best throughput when the packet size is very small (50 bytes). This is because the overhead of RTS/CTS or PION transmissions cannot be offset by their benefit in preventing hidden terminal transmissions. An interesting observation is that IEEE 802.11 without RTS still has better throughput than does 802.11 with RTS with the large packet sizes, which is against the general belief that RTS/CTS can help improve the MAC efficiency when the packet size is large. This is because in the chain scenario, all the packets are being sent towards the same direction, so that a transmission at a downstream hop may cause the transmission at the immediate upstream hop to fail but not vice versa. Further, when such a hidden downstream terminal causes some packet drops at the upstream, fewer packets from upstream nodes survive to be received by downstream nodes, limiting the number of further such collisions that can be caused by transmissions from these hidden downstream nodes since they will finally get no packets to send. Therefore, the negative impact of hidden terminals in the chain scenarios is somewhat self-constrained, making IEEE 802.11 with RTS not able to offset the overhead from its RTS and CTS transmissions.

C. Evaluation in Cross Scenarios

We also evaluated EMAC in the cross scenarios with backlogged UDP loads; the path length of the flows is varied from 2 hops to 14 hops.

Figure 8 shows the average end-to-end throughput combined from the two flows in cross scenarios with increasing path length when the packet size is 1500 bytes. IEEE 802.11 without RTS performs the worst among the three protocols. This is because both EMAC and IEEE 802.11 with RTS use

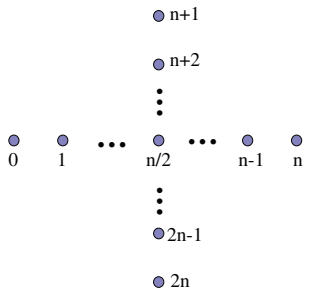

Fig. 4. Cross scenario

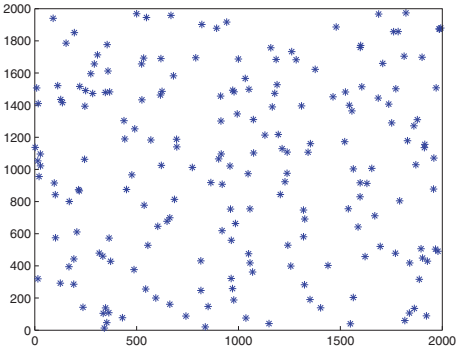

Fig. 5. A realistic 200-node network

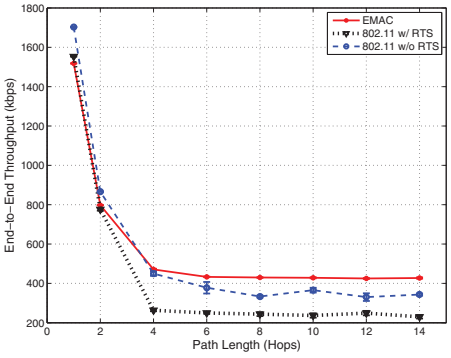

Fig. 6. Average end-to-end throughput in chain scenarios.

control frames to avoid the potential data frame transmission collisions caused by hidden terminals in the crossing area.

In Figure 8, an interesting observation is that when the path length of the flows is 4 hops, IEEE 802.11 receives almost zero throughput. In a 4-hop cross scenario using IEEE 802.11, when the two hop-1 nodes of each flow try to deliver their packets to their next-hop nodes, which are actually the very node at the crossing point, their packets have a very high chance to collide with each other. This problem is becoming worse in that while the two hop-1 nodes wait for each other or wait for the node at the crossing point to resolve their contention, their upstream nodes, the two hop-0 (source) nodes, which can only see clear medium, are still transmitting more packets to them, which further makes the contention at the crossing area worse. The intra-flow contention and inter-flow contention are both very high in the 4-hop cross scenario, driving the throughput of IEEE 802.11 down to a starvation level. For EMAC, however, since a PION frame can be potentially transmitted across multiple hops in one transaction, the source node and the node at the crossing point can coordinate to reduce the severe intra-flow and inter-flow contention; EMAC is thus still be able to have good throughput in this case.

D. Evaluations in a Realistic Scenario

We also evaluated EMAC in the 200-node realistic random scenario (Figure 5) with backlogged UDP loads. Traffic loads in the simulations come from a number of randomly selected 12-hop backlogged UDP flows. The number of UDP flows was varied from 1 to 8, resulting increased inter-flow contention as more flows are added in the network.

Figure 9 shows the change in average combined end-to-end throughput from all flows in the realistic scenario when the packet size is 1500 bytes. IEEE 802.11 without RTS has the worst performance among the three protocols when the number of flows in the network is more than 4 due to its lack of protection against hidden terminals. IEEE 802.11 with RTS, however, has the worst performance among the three protocols when the number of flows in the network is within 4. This is because the use of RTS/CTS frames to protect data frame transmissions against hidden terminals becomes less necessary when inter-flow contention is not severe. Among the three protocols, EMAC generally shows the best performance, regardless of the number of flows.

IV. Related Work

Many MAC protocols have been proposed for multi-hop wireless networks, including many contention-free protocols and multi-channel protocols [6]. We discuss here only contention-based single-channel protocols, the type of protocol to which both IEEE 802.11 DCF and EMAC belong.

The limitation of IEEE 802.11 DCF in multi-hop wireless networks has long been recognized [2], [3]. When the traffic load is high, the CSMA/CA random access mechanism drops its efficiency significantly [7].

In order to improve the performance of CSMA/CA mechanism in multi-hop wireless networks, several schemes have been proposed to reduce the overhead of multiple CSMA/CA transactions by combining two or more frames (from different transactions) into a single frame, similar with what EMAC does in assigning a PION as both an RTS and CTS.

Choudhury et al. proposed a scheme to combine a CTS and an ACK [8]. When a node receives multiple packets from the same source, the ACK of a previously received data frame can be piggybacked in a CTS frame of the later request. Their optimization is only for the single-hop networks and is not able to address the problem of multi-hop traffic. Berger et al. proposed a scheme called quick-exchange (QE) for piggybacking an ACK within a data frames if the ACK sender happens to have a data frame waiting to go to the ACK receiver [9]. However, the combined ACK/DATA frame may suffer the hidden terminal collisions as the ACK receiver did not broadcast its earlier RTS with the knowledge that an extra DATA would come with the ACK. Several schemes have been proposed to combine an ACK of one transaction and a RTS of a later transaction. DCF+ is designed for single-hop wireless networks only [10]. Data-driven cut-through multiple access (DCMA) and fast forward (FF) both combine the ACK for the previous hop and the RTS for the next hop if the data being acknowledged needs to be relayed further downstream [11], [9]. Queue-driven cut-through medium access (QCMA) is slightly different from DCMA and FF in that the piggybacked RTS is used for the current data in queue, not necessary the one that is being acknowledged [12]. Compared with these

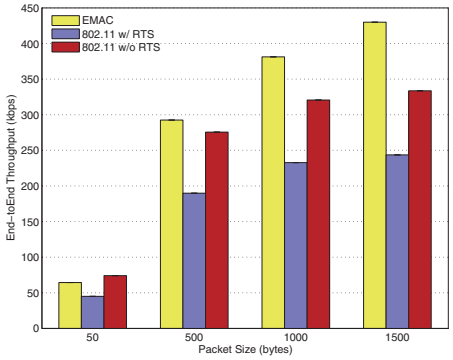

Fig. 7. End-to-end throughput with different packet sizes in the 8-hop chain scenario

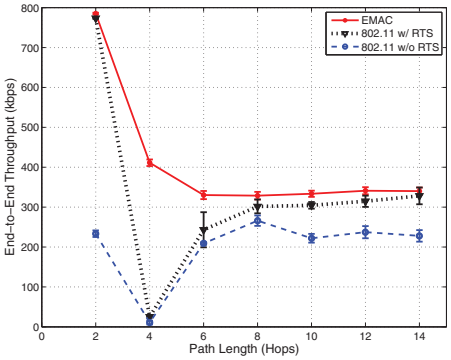

Fig. 8. Average end-to-end throughput in the cross scenarios.

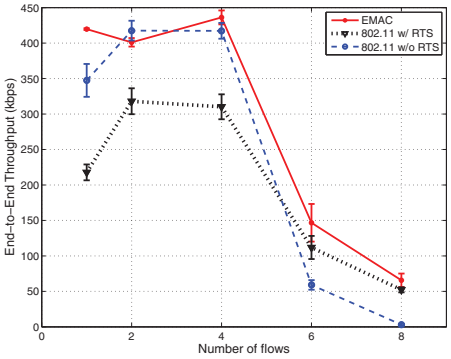

Fig. 9. Combined average end-to-end throughput in the realistic scenario.

schemes, the method of EMAC in combining an RTS and a CTS is unique. Also, EMAC can start transmitting the data frame at the upstream while the scheduling PION frame is still being transmitted at the downstream; therefore, EMAC can potentially have better temporal and spacial efficiency.

Toh et al. proposed a scheme called MARCH to utilize a CTS-only handshake in multi-hop wireless networks [13]. MARCH operates in a similar way to a receiver-based MAC protocol as it requires the downstream node to proactively send a CTS to the upstream node to request for data. MARCH therefore will not work well if the nodes in the network have different views on route selection. Carlson et al. proposed an algorithm called DARE that does a round-trip end-to-end reservation using control frames prior to actual traffic [14]. The control frame can also play two roles at a time in DARE as PION does. However, DARE is used only in handling periodic traffic that has real-time requirement. For unpredictable traffic, DARE's round-trip reservation scheme will introduce large overhead and long delay to data transmission.

Our prior work of RMAC also combines RTS and CTS into a single PION frame to achieve low delivery latency in wireless sensor networks [4]. However, RMAC works only with *synchronous* duty-cycling networks, where all nodes have synchronized clocks to switch their radio on or off at the same time, and traffic load is very light. These conditions are not assumed by EMAC, which can be applied to general always-on *asynchronous* networks.

V. Conclusions and Future Work

In this paper, we have introduced a new wireless MAC protocol, EMAC, to improve the efficiency of medium reservation for asynchronous multi-hop wireless networks such as mobile ad hoc networks (MANETs) and wireless mesh networks. In EMAC, synchronized multi-hop medium reservation can be achieved over distributed asynchronous nodes through a Pioneer (PION) scheduling frame and distributed schedule resolution algorithm. Simulation results under various scenarios and traffic loads have shown the great potential of EMAC's PION mechanism over RTS/CTS mechanism in improving throughput in asynchronous multi-hop wireless networks. For example, in chain scenarios, EMAC improves throughput over IEEE 802.11 with RTS/CTS by up to 85%. In cross scenarios, EMAC also avoids starvation problem occurred to IEEE 802.11.

Despite the potential that EMAC has shown in this work, there are some issues left open for future investigations. For example, the PION mechanism increases the complexity in packet handling due to its cross-layer design, which may demand for more memory and MIPS power in a real implementation. Also, the interactions between the EMAC and TCP protocols could be an interesting topic for future study.

References

[1] IEEE Computer Society, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std 802.11-1997, 1997.

[2] S. Xu and Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless adhoc networks?" *IEEE Communications Magazine*, vol. 39, no. 6, pp. 130–137, Jun. 2001.

[3] H.-Y. Hsieh and R. Sivakumar, "IEEE 802.11 over multi-hop wireless networks: Problems and new perspectives," in *IEEE 56th VTC*, 2002.

[4] S. Du, A. K. Saha, and D. B. Johnson, "RMAC: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks," in *INFOCOM 2007*, May 2007.

[5] S. Du, "Using routing information to improve MAC performance in multi-hopwireless networks," Ph.D. dissertation, Rice University, 2008.

[6] S. Kumar, V. S. Raghavan, and J. Deng, "Medium access control protocols for ad hoc wireless networks: A survey." *Ad Hoc Networks*, vol. 4, no. 3, pp. 326–358, 2006.

[7] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *ACM MobiCom 2001*, Jul. 2001, pp. 61–69.

[8] R. R. Choudhury, A. Chakravarty, and T. Ueda, "Implicit MAC acknowledgment: An optimization to 802.11," in *WTS 2005*, Apr. 2005.

[9] D. Berger, Z. Ye, P. Sinha, S. Krishnamurthy, M. Faloutsos, and S. K. Tripathi., "TCP-friendly medium access control for ad-hoc wireless networks: Alleviating self-contention." in *IEEE MASS 2004*, Oct. 2004.

[10] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of reliable transport protocol over ieee 802.11 wireless LAN: Analysis and enhancement," in *INFOCOM 2002*, pp. 599–607.

[11] A. Acharya, A. Misra, and S. Bansal, "A label-switching packet forwarding architecture for multi-hop wireless LANs," in *ACM WoWMoM 2002*, 2002.

[12] D. Raguin, M. Kubisch, H. Karl, and A. Wolisz, "Queue-driven cut-through medium access in wireless ad hoc networks," in *IEEE WCNC 2004*, Mar. 2004.

[13] C.-K. Toh, V. Vassiliou, G. Guichal, and C.-H. Shih, "MARCH: a medium access control protocol for multihop wireless adhoc networks," in *IEEE MILCOM 2000*, Oct. 2000.

[14] E. Carlson, C. Prehofer, C. Bettstetter, H. Karl, and A. Wolisz, "A distributed end-to-end reservation protocol for IEEE 802.11-based wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2018–2027, 2006.

978-1-4244-5638-3/10/$26.00 ©2010 IEEE