Proceedings of the 25th IEEE International Conference on Advanced Information Networking and Applications Workshops, AINA 2011
Biopolis, Singapore, March 22-25, 2011
IEEE Computer Society 2011

A Medium Access Control Protocol with Adaptive Parent Selection Mechanism for Large-Scale Sensor Networks

Behnam Dezfouli [†1], Marjan Radi [†2], Mohammad Ali Nematbakhsh [‡3], Shukor Abd Razak [†4]
[†] Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia (UTM), Johor, Malaysia
[‡] Faculty of Computer Engineering, University of Isfahan, Iran
[1]dezfouli@ieee.org, [2]radi@ieee.org, [3]nematbakhsh@eng.ui.ac.ir, [4]shukorar@utm.my

***Abstract*—In the MAC protocols based on the S-MAC scheme, usually the combination of periodic sleep/listen scheduling and four-way handshake mechanism is employed to reduce idle listening and avoid interference. However, this combination greatly degrades network capacity and results in high end-to-end latency. In this paper, we propose Adaptive IAMAC to increase channel utilization and improve communication efficiency, specifically in large-scale sensor networks with low duty cycle. Adaptive IAMAC allows multiple nodes to transmit to their common parent during a frame. Moreover, it includes the adaptive parent selection mechanism, which enables the nodes to change their parent according to the currently overheard control packets at the MAC layer. Through these techniques, Adaptive IAMAC enhances network throughput, reduces end-to-end latency, and moderates the overhead of four-way handshake mechanism. Simulation results confirm that Adaptive IAMAC provides significant improvements over S-MAC in terms of throughput, latency, and energy efficiency.**

Keywords-Adaptive IAMAC; Sleep/Listen Scheduling; Tree Routing; Contention-Based; Latency; Lifetime.

I. INTRODUCTION

Radio communication is the major source of energy consumption in wireless sensor networks. Since the radio operation is controlled by MAC layer, significant lifetime improvement can be achieved through MAC protocol optimization. Furthermore, due to the short-range radio communication and necessity of multi-hop packet transmission in wireless sensor networks, MAC protocol highly affects on the end-to-end latency of packets.

TDMA MAC protocols arbitrate medium access through slot assignment [1][2][3]. Because each node is allowed to transmit in its dedicated slots, TDMA scheme efficiently avoids inter-node interference and increases channel utilization in high traffic loads [4][3]. The major drawback regarding to the TDMA protocols is that they incur costly signaling overhead to perform time synchronization and update slot assignment. More importantly, signaling overhead increases as the network size grows; therefore, these techniques cannot be used in large-scale sensor networks. Alternatively, the periodic sleep/listen scheduling does not need fine-grained time synchronization and is a simple yet efficient solution to reduce duty cycle. The MAC protocols based on the S-MAC [5] scheme usually employ a combination of sleep/listen scheduling and contention-based access method to achieve both low duty cycle and contention resolution. However, because a pure CSMA access method does not behave well under heavy traffic loads, the four-way handshake mechanism (i.e., RTS/CTS/DATA/ACK) is typically adopted to reduce interference through channel reservation [5][6][7]. There are two major problems concerning these MAC protocols. First, combination of the four-way handshake mechanism with periodic sleep/listen scheduling significantly reduces network capacity and results in high end-to-end latency [8][5]. Second, given the small packet size in wireless sensor networks, this handshake mechanism incurs very high overhead in terms of energy consumption and channel utilization [9][10].

In this paper, we propose Adaptive IAMAC (Adaptive Interference Avoidance MAC), which integrates the adaptive parent selection mechanism and interference avoidance as a periodic sleep/listen MAC protocol. Since convergecast is the main communication pattern observed in sensor networks, Adaptive IAMAC benefits from the tree-based routing scheme and introduces a new RTS/CTS handshake mechanism, which allows multiple nodes to transmit to their common parent during a frame (a frame is a complete cycle of listen and sleep). Additionally, through interaction with the network layer, and according to the currently overheard control packets at the MAC layer, Adaptive IAMAC enables the nodes to change their parent adaptively. Besides, Adaptive IAMAC supports early node deactivation whenever a node cannot participate in any communication or when its participation may result in inter-node interference. Through these mechanisms Adaptive IAMAC increases channel utilization for the four-way handshake scheme, reduces multi-hop latency, and provides higher energy efficiency.

Supporting multiple transmissions to a parent node during a frame is also proposed in IAMAC [11]. Although IAMAC interacts with the network layer to support multiple transmissions and avoid interference, however, it cannot affect on the path selection mechanism so that parent nodes are merely determined by the routing protocol. In contrast, Adaptive IAMAC introduces a joint MAC/network layer parent selection mechanism that enables the nodes to decide about their next-hop node at the start of each frame. The adaptive parent selection mechanism determines the next-hop node according to the neighbor table entries and newly exchanged control packets perceived at the MAC layer. This mechanism further reduces the overhead of exchanging control packets, increases channel utilization, and results in lower end-to-end latency compared to IAMAC. Through simulation we show that Adaptive IAMAC outperforms S-MAC [5] and IAMAC in terms of throughput and latency,

while it also preserves its energy efficiency.

The rest of this paper is organized as follows: In Section II we provide an overview over the previously proposed MAC protocols. Section III presents the design of Adaptive IAMAC. We perform performance evaluation in Section IV. We conclude in Section V.

II. Related Work

S-MAC [5], T-MAC [7], and Adaptive S-MAC [6] utilize the RTS/CTS handshake mechanism for collision avoidance and employ periodic sleep/listen scheduling to reduce idle listening. Because the RTS/CTS exchange for channel reservation alleviates the need for tight time synchronization, these protocols can easily achieve their required time synchronization through broadcasting beacon messages at certain intervals. T-MAC proposes the adaptive adjustment of listen duration to further reduce the idle listening time of S-MAC. However, both S-MAC and T-MAC demonstrate high end-to-end latency due to using RTS/CTS handshaking and employing low duty cycle. Whenever a node overhears a RTS or CTS packet, it cannot contend for channel access until the start of the next frame. Hence, when several nodes are in the carrier sensing range of each other and contend for medium access, only one node can transmit its data packets during a frame. Due to the high latency of S-MAC, Adaptive S-MAC [6] proposes the coordinated adaptive sleeping mechanism, which allows a data packet to be moved more than one hop (about two hops) during a frame. Whenever a node overhears a RTS or CTS packet, it should wake up for a short duration during its sleep period to receive probable packets. With this mechanism, if the neighboring node wakes up while it is not the next-hop node, it incurs energy waste due to overhearing or idle listening. According to the evaluations presented in [11], Adaptive S-MAC demonstrates significant reduction in network lifetime as the neighborhood size increases.

Several TDMA [1][2] and hybrid TDMA/CSMA [3][4] techniques are proposed to improve communication efficiency, especially for heavy load conditions. In TRAMA [2] collision-free schedules are assigned to the nodes according to the traffic demands. Although TRAMA provides high channel utilization, its delay is worse than S-MAC due to the scheduling overhead. In addition, its computational complexity and signaling overhead may limit its implementation in large-scale sensor networks. Z-MAC [3] needs to periodically run its slot assignment algorithm (DRAND) to preserve high throughput. However, due to its overhead, DRAND should not be run periodically and authors recommend its execution only at the network initialization. The testbed results provided in [4] (for two different testbeds with 45 and 31 Mica2 nodes), confirm that Z-MAC throughput degrades as time proceeds. The two protocols TreeMAC [1] and DMAC [10] rely on the tree-like communication pattern to perform their slot assignment. DMAC [10] tries to reduce the latency of S-MAC through assigning active periods in a tree-like fashion. Despite the dependency of DMAC on predetermined routes, no route adaptation mechanism is provided for this protocol. In addition, packet corruption may also be a problem in high traffic loads since collision avoidance methods are not utilized. TreeMAC [1] highly depends on the time synchronization accuracy and performs its slot assignment from the root towards the leaves; therefore, it may not be possible to use this protocol for large-scale sensor networks. In both TreeMAC and DMAC, child-parent transmission is the only communication pattern provided. Accordingly, these protocols do not support peer-to-peer data transmission among the neighboring nodes, which is especially useful for in-network processing and data aggregation.

As a conclusion, TDMA MAC protocols rely on the frequent exchange of control packets due to the changes in traffic pattern, routing path, and node synchronization. In fact, TDMA MAC protocols are appropriate when the traffic load is high and nodes exhibit a similar traffic pattern. In contrast, contention-based MAC protocols provide higher adaptability to topology changes and traffic variations in large-scale sensor networks [12]. However, their performance highly depends on the efficiency and overhead of underlying collision avoidance mechanism.

III. Design of Adaptive IAMAC

A. Layers and Data Structures

The two data structures at the network layer are *Packet Queue* and *Neighbor Table*. Packet Queue includes the data packets that should be delivered to the next-hop node. Each entry in the Neighbor Table corresponds to a neighboring node and its cost to the sink. We assume after neighbor discovery each node knows its neighbors and their ETX cost [14] towards the sink. The neighbor node with minimum ETX cost is called *BestParent*. The MAC layer includes the *RTS Queue*. This data structure is used to record the received RTS packets during a frame. The two data structures of the network layer should be accessible to the MAC protocol.

B. Sleep/Listen Scheduling Structures

The two periodic sleep/listen structures are demonstrated in Figure 1. Since underlying time synchronization protocol affects on the synchronization rate, two sleep/listen structures are presented to make the synchronization independent from sleep/listen scheduling. Suppose that SI denotes the required synchronization interval that satisfies synchronization accuracy. When the time interval between two successive *RTS Slots* is less than SI, the *Time Frame Structure* (Figure 1(a)) can be applied. Otherwise, when the time interval between two successive RTS Slots is longer than SI, the *Super Frame Structure* (Figure 1(b)) is appropriate. Notice that in order to preserve time synchronization, the duration between two consecutive *Sync/Routing Slots* should not exceed the SI interval. The choice between these two structures mainly depends on the application demands. Using the Super Frame Structure, duty cycle can be highly reduced to increase network lifetime. In contrast, the Time Frame Structure provides lower latency due to its shorter frame duration.

For brevity, in the rest of this paper we use the term *frame* instead of “Time Frame Structure” and “Super Frame Structure”. We assume SI equal 12 seconds and whenever

the frame duration is longer than 12 seconds Super Frame Structure is used and whenever the frame duration is less than 12 seconds Time Frame Structure is applied.

In both scheduling structures, a frame includes a *Sync/Routing Slot*, *RTS Slot*, *CTS Slot*, and *Sleep/Communication Slot*. The time synchronization and routing information can be embedded in the *Sync/Routing packets* and should be broadcasted during the Sync/Routing Slot using the CSMA access method. Each node with a non-empty Packet Queue can contend for medium access during the RTS Slot to send a RTS packet to its parent node. A parent node sends a CTS packet to its children during the CTS Slot. Data transmission can be performed during the Sleep/Communication Slot.

C. Channel Access Mechanisms

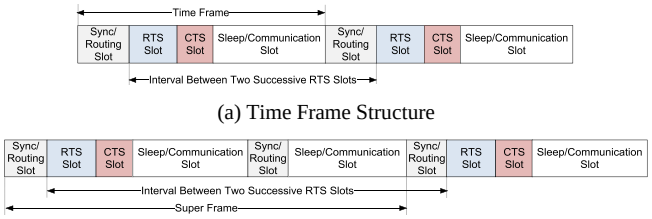
In order to allow multiple nodes send their data packets to their common parent during a frame, they should be able to send their RTS packets to the common parent during a RTS Slot. However, children of a node may not be in the carrier sensing range of each other and their RTS packets may collide at the parent (hidden terminal problem). Figure 2(a) shows the channel access mechanism during the RTS Slot. The RTS Slot is divided into several slots, each one called a *RTS Contention Slot*. The duration of a RTS Contention Slot is equal to the transmission time for a RTS packet plus the contention window period. The number of RTS Contention Slots can be calculated using the network density and traffic rate.

When a node arrives at the RTS Slot and its Packet Queue is not empty, it randomly selects a RTS Contention Slot and chooses a random backoff time to perform clear channel assessment (CCA) during the selected RTS Contention Slot. For example, in Figure 2(a) when the node arrives at the fourth RTS Contention Slot, it backoffs for a random duration and transmits a RTS packet when the backoff timer reaches zero and channel is free. Notice that all the nodes (with empty or non-empty Packet Queue) should listen to the channel during the contention window of each RTS Slot; however, some nodes may become deactivated before the end of the RTS Slot. In this paper, we consider five RTS Contention Slots with contention window equal 15 slots.

In the CTS Slot, parent nodes multicast a CTS packet to their children (a parent node is the node with non-empty RTS Queue). The channel access mechanism during the CTS Slot is demonstrated in Figure 2(b). Given that the contention intensity for CTS transmission is much lower than the contention for RTS transmission, CTS Slot is not divided into sub slots. Accordingly, the CTS Slot duration equals to the transmission time for a CTS packet plus the contention window duration. The parent nodes perform CCA for a random duration before transmitting CTS packet to their children. The contention window size for CTS Slot is 15 slots.

D. Algorithms

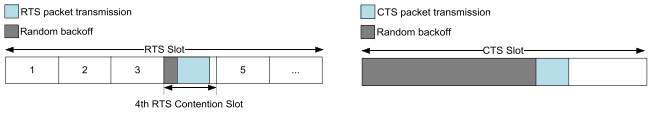
Adaptive IAMAC presents two algorithms. The first algorithm runs in the RTS Slot, and the second algorithm runs in the CTS Slot.



(a) Time Frame Structure

(b) Super Frame Structure

Figure 1. Sleep/listen scheduling structures.



(a) Channel access mechanism during RTS Slot

(b) Channel access mechanism during CTS Slot

Figure 2. Channel access mechanisms.

Figure 3 demonstrates the flowchart of the RTS Slot algorithm. Because it is not feasible to present all the possible scenarios, we demonstrate a fairly simple scenario to clarify the operation of this algorithm. This scenario is presented in Figure 4. The time index below each subfigure (Time x) indicates the time progress and each one corresponds to a different RTS Contention Slot. Notice that these time steps are not necessarily consecutive RTS Contention Slots; however, they all belong to the same RTS Slot. For each time index, the details beside each node represent the status of that node after the corresponding transmissions are completed. It is assumed that Packet Queues of node E, D, B, and C are nonempty. RTSQ stands for the RTS Queue. Each RTS packet includes receiver address, sender address, and the estimated transmission duration, which can be calculated using the link quality and number of data packets in the Packet Queue. The two control variables (*CancelRTSTrans* and *CancelCTSTrans*) are mainly used to decide about the role change (i.e., sender/receiver) and early node deactivation. Note that a deactivated node wakes up again at the start of the next frame.

At Time 1, node A and B receive a RTS packet from node E and C, respectively. Additionally, node D overhears the RTS packet transmitted from E to A; therefore, it extracts the destination address included in the overheard RTS packet and tries to find a match in its Neighbor Table. If a match is found and that is a *qualified neighbor*, it can be selected as the new parent. The qualified neighbor is defined as follows: Neighbor *i* is a qualified neighbor if it satisfies the following inequality,

$$Cost_i \le (1+\rho) \cdot Cost_{BestParent} \qquad (1)$$

where $Cost_i$ is the cost of node *i*, $Cost_{BestParent}$ is the cost of the BestParent (BestParent is the default parent node selected by the routing algorithm as described in Section III.A), and ρ is a constant (the ρ value is 0.2 in our simulations unless

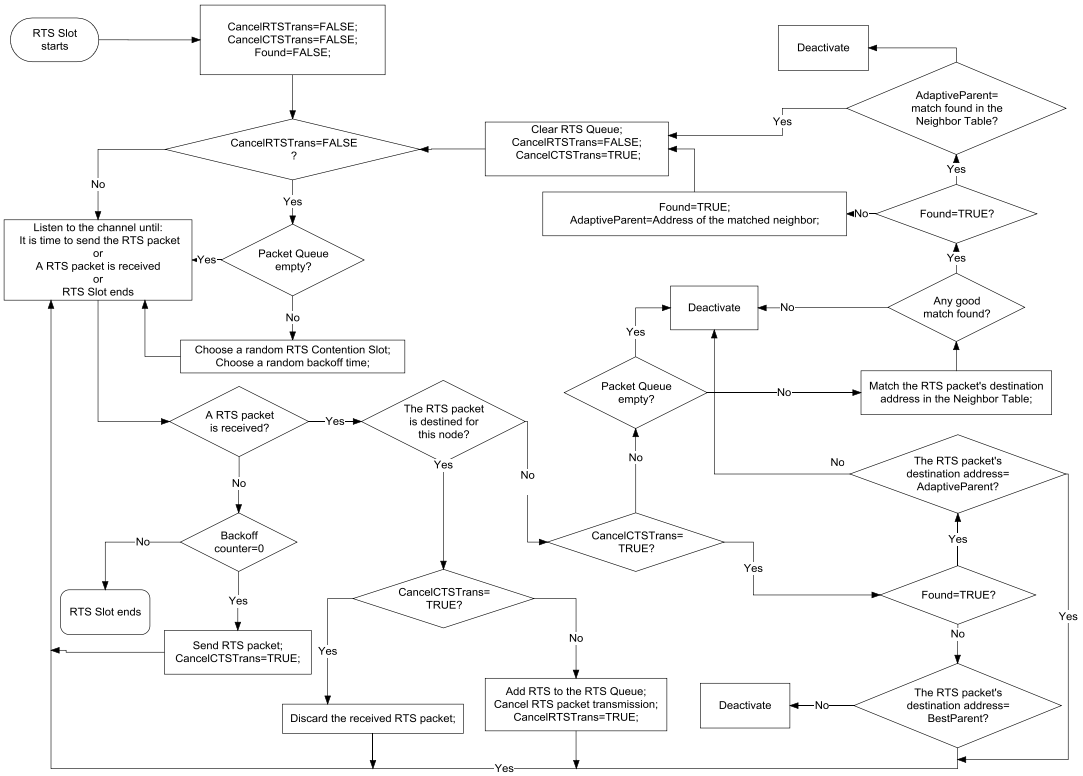


Figure 3. Flowchart of the RTS Slot's algorithm. *BestParent* is the default parent node selected by the routing algorithm. *AdaptiveParent* is the parent node selected by the adaptive parent selection mechanism.

otherwise specified). Suppose that node D has previously selected node B as its BestParent. Since node D finds a match for node A in its Neighbor Table and node A is a qualified neighbor for node D, node D selects node A as its *AdaptiveParent* for the current frame. In addition, the new values of control variables still allow node D to send a RTS packet.

At Time 2, node A receives a RTS packet from node D, while B and E overhear this packet. When node B overhears this packet, it understands that it cannot receive data packets from node C due to probable packet collision with node D. Therefore, node B checks its Neighbor Table to see if it can select node A as its new parent (notice that node B's BestParent is node X). As it can be observed from the figure, node B changes its parent, adjusts its control variables, and flushes its RTS Queue because it decides to act as a sender not a receiver (role changes). When node E overhears the RTS packet transmitted from D to A, it does not change its control variables because the destination address of the overheard RTS packet matches with the node to which node E has previously transmitted its RTS packet.

At Time 3, node A receives a RTS packet transmitted from node B. When node D overhears this packet, it does not change its control variables because the destination address included in the overheard RTS packet is the same as the node D's currently selected parent (i.e., node A). Node C is deactivated because it has previously transmitted a RTS packet to the node that is different from the destination address of the overheard RTS packet.

At the end of the RTS Slot, node A includes the RTS packets received from node E, D, and B. According to the estimated transmission durations included in these RTS packets, node A sends a multicast CTS packet containing the activation time for each child node. The CTS packet must be transmitted during the CTS Slot with the channel access method described in Section III.C.

Since the CTS Slot's algorithm is rather simpler than the RTS Slot's algorithm, we provide the pseudo code of this algorithm in Algorithm 1.

IV. Performance Evaluation

The simulation application is programmed in OMNeT++ framework using the characteristics of Mica2 motes and the data link model of [13]. Table 1 represents the general simulation settings. We assume that all the nodes sample the environment periodically. Each sample corresponds to a data packet with 29 bytes payload size.

A. Latency

Latency of a packet is defined as the time duration from packet generation at the source node to its reception at the sink node. Figure 5 demonstrates the average latency of the packets received at the sink node. As it can be observed, both Adaptive IAMAC and IAMAC demonstrate significantly lower end-to-end latency than S-MAC. Although in the Adaptive IAMAC, IAMAC, and S-MAC a packet can be

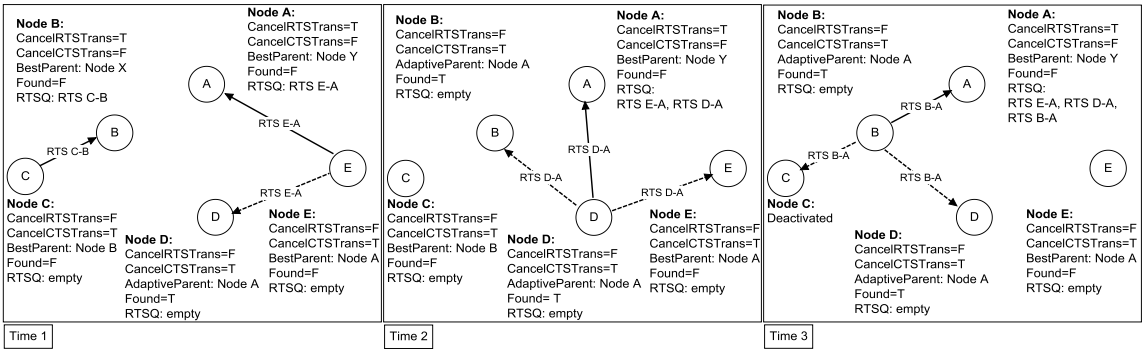


Figure 4. A sample scenario for RTS Slot.

Algorithm 1. CTS Slot's algorithm.

```
1.  /*CancelRTSTrans and CancelCTSTrans variables are defined
    in the RTS Slot's algorithm*/

2.  If (RTSQueue.Length!=0)
3.      Choose a random backoff time;

4.  While (CTS Slot is not finished)
5.  {
6.   //when a CTS packet arrives
7.   If (a new packet is received)
8.      Pkt=Arrived Packet;

9.   /*if a CTS packet is overheard, cancel CTS transmission*/
10.  If ( (Pkt.DestinationAddress!=MyAddress) || (channel busy) )
11.     Deactivate the node;

12.  If (backoff counter==0)
13.     //multicast a CTS packet to children
14.     Send CTS packet;

15.  /*if a CTS packet is received, data packets can be transmitted
     during the Sleep/Communication Slot*/
16.  If (Pkt.DestinationAddress==MyAddress)
17.     Extract the timing information;
18.     Set a timer to wake up during the Sleep/Communication Slot;
19.     Sleep;
20. }
```

moved at most one hop during a frame, Adaptive IAMAC and IAMAC allow multiple nodes to transmit to their common parent during a frame. Moreover, Adaptive IAMAC reduces the end-to-end latency of IAMAC through the adaptive parent selection mechanism. For the similar frame durations, the gain of Adaptive IAMAC over S-MAC and IAMAC is about 90% and 30%, respectively. Notice that even if we increase the frame duration for Adaptive IAMAC, it still indicates lower latency than S-MAC. For example, the latency improvement of Adaptive IAMAC (10 sec) and Adaptive IAMAC (15 sec) over S-MAC (5 sec) are about 80% and 60%, respectively.

Since the protocols evaluated in Figure 5 employ the periodic sleep/listen scheduling, each packet experiences a minimum latency on its path towards the sink. The minimum latency depends to the number of hops and frame duration. Consequently, as the traffic rate descends (i.e., sampling interval increases), the slope of the curves reduces and the latency of each protocol tends to its almost fixed value.

B. Lifetime

Figure 6 demonstrates the average lifetime of the nodes. Although for the same frame duration Adaptive IAMAC provides lower lifetime than S-MAC, nevertheless, Adaptive IAMAC provides higher performance than S-MAC in both lifetime and latency. This is due to the substantially reduced latency of Adaptive IAMAC even if its frame duration is longer than that of S-MAC. For example, Adaptive IAMAC (10 sec) provides about 10% improvement in lifetime and 80% improvement in latency over S-MAC (5 sec). Also, the gains of Adaptive IAMAC (15 sec) over S-MAC (5) are 25% and 60% in terms of lifetime and latency, respectively.

Figure 6 also confirms the very low lifetime of Adaptive S-MAC, which is due to its coordinated adaptive sleeping mechanism. Compared to Adaptive S-MAC (5 sec), Adaptive IAMAC (5 sec) provides 220% improvement in the lifetime.

Simulation results show that for the similar frame durations, the lifetime of Adaptive IAMAC is about 8% less than the lifetime of IAMAC. This lower lifetime is caused by the adaptive parent selection mechanism, which allows the neighboring nodes other than the BestParent to be selected as the next-hop node. In fact, through selecting parent nodes with higher cost, Adaptive IAMAC trades energy for latency.

The lifetime of Adaptive IAMAC mainly depends on the following factors: 1) traffic generation rate, 2) frame duration, 3) number of supported transmissions per frame, and 4) number of deactivated nodes per frame. We provide an example to show the interdependency and effect of these factors on energy efficiency. According to the RTS Slot's algorithm (Figure 3), whenever a node overhears a RTS packet it should be deactivated if its Packet Queue is empty and it has not sent any RTS packet earlier. This early node sleeping depends on the frame duration and traffic generation rate. Considering T as the packet generation interval, nodes generate data packets at time t_1, $t_2=t_1+T$, and so on. We consider T big enough to satisfy $T>>f$, where f is the frame duration. Suppose that the adaptive parent selection mechanism enables node A (and its sibling nodes) to send its data packets to node B (parent node) at time t_x, where $t_1<t_x<t_2$. Since node A's Packet Queue is empty during t_2-t_x, therefore, node A should be deactivated upon overhearing a

TABLE 1. DEFAULT SIMULATION SETTINGS

Radio			
Modulation	FSK	Encoding	NRZ
Output Power	0 dBm	Bandwidth	19.2 Kbps
Transmission Medium			
Path Loss Exponent	4	PL_{D0}	55 dBm
Noise Floor	-105 dBm	D_0	1 m
Other Parameters			
Number of Nodes	200	Area	100×100 m^2
Packet Payload	29 B	Packet Size	45 B

RTS packet. Now, suppose that the adaptive parent selection mechanism is disabled and node A sends its data packets to node B at time t_y such that $t_1<t_x<t_y<t_2$. Accordingly, as t_2-t_x reduces to t_2-t_y, there will be fewer numbers of RTS Slots in which node A can be deactivated upon overhearing a RTS packet.

C. Throughput

In this section, we compare the channel utilization of Adaptive IAMAC, S-MAC, and IAMAC. Throughput is defined as the number of transmitted (and correctly received) data bits per second. Notice that we have also excluded MAC header and CRC bits in throughput calculation.

Figure 7 presents the throughput evaluation for two different packet generation intervals (the value in the parentheses indicates the packet generation interval). Although Adaptive IAMAC and IAMAC both support multiple transmissions to a common parent and show significant throughput enhancement compared to S-MAC, Adaptive IAMAC also utilizes the adaptive parent selection mechanism to boost throughput. Averaged over the four frame durations, Adaptive IAMAC (60 sec) improves the throughput of S-MAC (60 sec) and IAMAC (60 sec) for about 105% and 23%, respectively. Figure 7 shows that the performance gain of Adaptive IAMAC becomes more apparent as the frame duration increases (i.e., duty cycle reduces). For example, with 60 seconds sampling interval, the gain of Adaptive IAMAC over S-MAC is 82% for 5 seconds frame duration and 134% for 25 seconds frame duration. Similarly, with 150 seconds sampling interval, the gain of Adaptive IAMAC over S-MAC is 29% for 5 seconds frame duration and 91% for 25 seconds frame duration. This demonstrates the higher capacity of Adaptive IAMAC, which is particularly evident at low duty cycle and high traffic rate.

D. Flexibility of Adaptive Parent Selection and its Effects on Performance

Since the flexibility of the adaptive parent selection mechanism mainly depends on the ρ value, in Figure 8 and Figure 9 we varied the value of ρ in 0.1 intervals to see its effect on the throughput and lifetime. Figure 8 shows that throughput enhancement can be achieved through increasing the ρ value. Since the adaptive parent selection mechanism needs a node to be a qualified neighbor before being selected

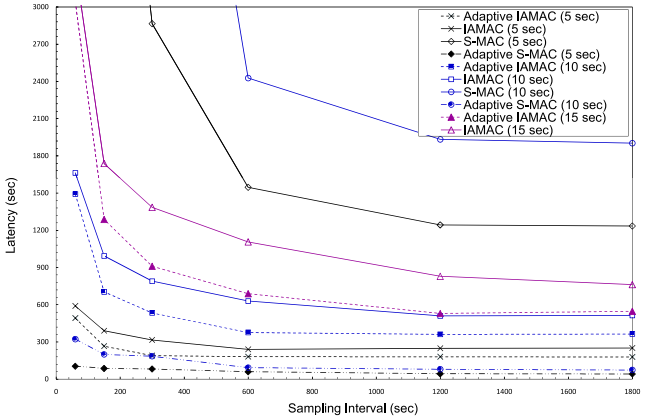


Figure 5. End-to-end latency comparison. The value in the parantheses indicates the frame duration.

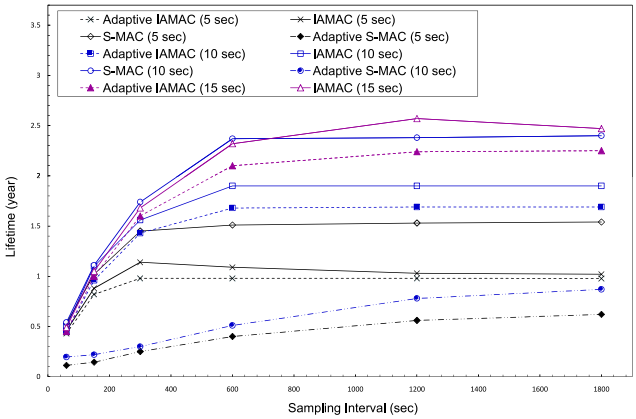


Figure 6. Lifetime comparison. The value in the parentheses indicates the frame duration.

as the parent, increasing the ρ variable improves the chance of adaptive parent selection through augmenting the number of qualified neighbors per node. However, according to the Inequality (1), this throughput improvement comes as the cost of lower energy efficiency due to selecting parent nodes with higher cost (AdaptiveParent can have higher cost compared to the BestParent). Figure 9 shows how increasing the ρ value (and transmitting data over lower quality links) affects on the network lifetime.

Given that in wireless sensor networks packets may be received through low quality links [13], Neighbor Table may include links with high packet corruption rate. Through blacklisting and Inequality (1) Adaptive IAMAC avoids the high-cost neighbors to be selected as the parent. The main objective of blacklisting is to select the m low-cost neighbors and record them as the m tuples of Neighbor Table [14]. Using Inequality (1), the Neighbor Table is further filtered through defining the set of qualified neighbors. For large ρ values, all the nodes in the Neighbor Table are qualified neighbors, therefore, the adaptive parent selection mechanism selects parent nodes regardless to their cost. In this paper we considered ten entries for the Neighbor Table (i.e., m=10), consequently, the adaptive parent selection

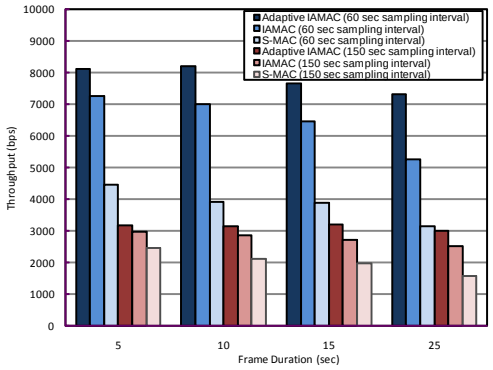


Figure 7. Effective throughput.

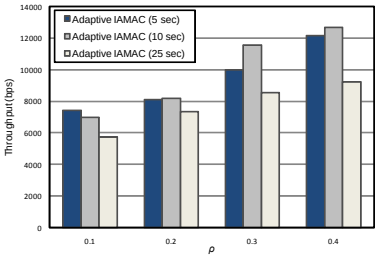


Figure 8. Effective throughput of Adaptive IAMAC vs. ρ value. Sampling interval is 60 seconds. The value in the parentheses indicates the frame duration.

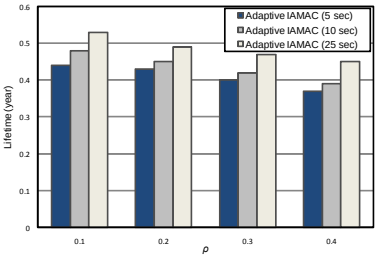


Figure 9. Lifetime of Adaptive IAMAC vs. ρ value. Sampling interval is 60 seconds. The value in the parentheses indicates the frame duration.

mechanism is limited to select from the ten best neighbors even when the ρ value is large. However, given that the neighboring nodes may demonstrate significant variations in their cost, it is recommended to utilize both blacklisting and Inequality (1). The optimum number of entries for the Neighbor Table can be determined using network density and transmission power. Then, the ρ value can be selected based on the lifetime and latency demands.

V. CONCLUSION

Motivated by the low throughout and high latency of periodic sleep/listen MAC protocols, we introduce Adaptive IAMAC, which is particularly suitable to provide high lifetime and low latency in large-scale sensor networks. Adaptive IAMAC 1) enables multiple nodes to transmit to their common parent during a frame, 2) allows the nodes to change their parent (i.e., next-hop node towards the sink) adaptively, and 3) supports early node deactivation. Specifically, the adaptive parent selection mechanism increases the number of transmissions per frame by enabling the MAC protocol to decide about the next-hop node according to the currently overheard control packets. In fact, through increasing the number of transmissions per frame, Adaptive IAMAC provides higher network utilization compared to other sleep/listen MAC protocols, especially when the duty cycle is low. Besides, in contrast to other tree-based MAC protocols, Adaptive IAMAC also permits peer-to-peer data transmission to support data aggregation. The performance evaluation results confirm that the proposed protocol provides considerable improvements in latency, lifetime, and throughput compared to other sleep/listen MAC protocols.

REFERENCES

[1] W.Z. Song, R. Huang, B. Shirazi, and R. LaHusen, "TreeMAC: Localized TDMA MAC protocol for real-time high-data-rate sensor networks," *Pervasive and Mobile Computing*, vol. 5, 2009, pp. 750-765.

[2] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," *Wireless Networks*, vol. 12, Feb. 2006, pp. 63-78.

[3] I. Rhee, A. Warrier, M. Aia, J. Min, and M.L. Sichitiu, "Z-MAC: A hybrid MAC for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 16, 2008, pp. 511-524.

[4] G.S. Ahn, S.G. Hong, E. Miluzzo, A.T. Campbell, and F. Cuomo, "Funneling-MAC: a localized, sink-oriented mac for boosting fidelity in sensor networks," *Proceedings of SenSys '06*, Boulder, Colorado, USA: ACM Press, 2006, pp. 293-306.

[5] J. Heidemann and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," *Proceedings of INFOCOM '02*, New York, USA: IEEE, 2002, pp. 1567-1576.

[6] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 12, Jun. 2004, pp. 493-506.

[7] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," *Proceedings of SenSys '03*, Los Angeles, California, USA: ACM Press, 2003, pp. 171-180.

[8] J. Li, C. Blake, D.S.J. De Couto, H.I. Lee, and R. Morris, "Capacity of Ad Hoc wireless networks," *Proceedings of MobiCom '01*, Rome, Italy: ACM Press, 2001, pp. 61-69.

[9] A. Woo and D.E. Culler, "A transmission control scheme for media access in sensor networks," *Proceedings of MobiCom '01*, Rome, Italy: ACM Press, 2001, pp. 221-235.

[10] G. Lu, B. Krishnamachari, and C.S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for tree-based data gathering in sensor networks," *Wireless Communications and Mobile Computing*, vol. 7, Sep. 2007, pp. 863-875.

[11] B. Dezfouli, M. Radi, and S. Abd Razak, "A cross-layer approach for minimizing interference and latency of medium access in wireless sensor networks," *International journal of Computer Networks & Communications*, vol. 2, Jul. 2010, pp. 126-142.

[12] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, 2006, pp. 115-121.

[13] M.Z. Zamalloa and B. Krishnamachari, "An analysis of unreliability and asymmetry in low-power wireless links," *ACM Transactions on Sensor Networks*, vol. 3, Jun. 2007, p. 7.

[14] O. Gnawali, M. Yarvis, J. Heidemann, and R. Govindan, "Interaction of retransmission, blacklisting, and routing metrics for reliability in sensor network routing," *Proceedings of SECON '04*, Santa Clara, California, USA: IEEE, 2004, pp. 34-43.