Modeling and Performance Analysis for Duty-cycled MAC Protocols with Applications to S-MAC and X-MAC

Ou Yang, *Student Member, IEEE,* Wendi Heinzelman, *Senior Member, IEEE*

Ou Yang and Wendi Heinzelman are with the University of Rochester.

This work was supported in part by NSF #CNS-0448046.

Abstract

Wireless sensor networks are often battery-powered, and hence extending the network lifetime is one of the primary concerns in the ubiquitous deployment of wireless sensor networks. One approach to efficiently utilize the limited energy supplies of the sensors is to have the medium access control (MAC) protocol duty-cycle the sensors, periodically putting the sensors to sleep and waking them up to reduce idle listening, which is energy intensive. Among duty-cycled MAC protocols, some protocols are synchronized so that nodes wake up at the same time in each cycle, and other protocols are asynchronous, where nodes have arbitrary offsets to start their cycles. For protocol designers, it is important to understand which type of duty-cycled MAC protocol should be chosen (synchronized or asynchronous), as well as what values should be assigned to the protocol parameters under a given network scenario in order to achieve a desirable performance for throughput, delay or energy consumption. However, previous work to analyze the performance of different duty-cycled MAC protocols is either protocol-specific, or limited to one aspect of the performance metric. In this paper, we propose a Markov queuing model to analyze the throughput, delay and energy consumption of both synchronized and asynchronous duty-cycled MAC protocols with applications to S-MAC and X-MAC. Our contributions include: (1) proposing a Markov queuing model to describe the queuing behavior of both synchronous and asynchronous duty-cycled nodes, (2) modeling the queue dynamics and the stationary probability of packet transmissions for S-MAC, a synchronized duty-cycled MAC protocol, to analyze its performance, (3) modeling the queue dynamics and the stationary probability of packet transmissions for X-MAC, an asynchronuous duty-cycled MAC protocol, to analyze its performance, (4) providing comprehensive performance estimation and comparison for different duty-cycled MAC protocols, and (5) providing flexibility to trade off different performance metrics by optimizing the protocol parameters. Our model results are validated by comparing with NS-2 and Matlab simulations.

Index Terms

Wireless sensor networks, duty-cycled MAC, S-MAC, X-MAC, throughput, delay, energy.

I. Introduction

Wireless sensor networks have attracted much interest in both academia and industry due to their low cost, ease of deployment, and, most importantly, support for various applications ranging from military surveillance and emergency rescue to medical monitoring. However, energy constraints imposed by the battery-powered sensor nodes are a limiting factor, preventing the ubiquitous use of wireless sensor networks. As a result, much research in wireless sensor networks has focused on how to save energy and prolong the network lifetime [1][2][3].

In particular, the MAC protocol can put sensors to sleep periodically to reduce idle listening [4][5][6][3], which is energy intensive in wireless sensor networks. Among such duty-cycled MAC protocols, some of them synchronize all the nodes in the network, so that the nodes sleep and wake up at the same time. Since the receiver is ready to receive when a sender wakes up and has a packet to send, synchronization improves the communication efficiency. However, synchronization requires overhead at each node to exchange sleep-wake-up schedules with the nodes' neighbors. This overhead can be significant when the data traffic is light in the network. Example synchronized duty-cycled MAC protocols include S-MAC [3] and T-MAC [6].

On the other hand, some of the duty-cycled MAC protocols are asynchronous. Asynchronous MAC protocols also put sensors to sleep periodically. However, every node has an arbitrary offset to start its sleep-wake-up cycles. Therefore, the synchronization overhead is removed, but a sender with packets to send may have to delay the transmission until the receiver wakes up. Example asynchronous duty-cycled MAC protocols include X-MAC [5] and Spec-MAC [7].

Comparing synchronized and asynchronous duty-cycled MAC protocols, it is not obvious which one has a better performance. Additionally, for any duty-cycled MAC protocols, it is not obvious what are the optimal parameters for a given network scenario and data arrival rates. Moreover, the performance of a MAC protocol includes not only energy consumption but also throughput and delay. Hence, it is highly desirable to propose a general freamwork that can analyze the performance of both synchronized and asynchronous duty-cycled MAC protocols. Previous work has been done on performance analysis for a specific MAC protocol or on a specific performance metric, such as delay or energy consumption. However, none of this previous work generalizes both synchronized and asynchronous duty-cycled MAC protocols and handles the analysis of the throughput, delay and energy consumption simultaneously.

Extending our previous work [8][9], we model and analyze the performance of duty-cycled MAC protocols with applications to S-MAC (synchronized) and X-MAC (asynchronous). To handle different MAC protocols, our approach decouples the performance analysis problem into two parts. Specifically, we propose a Markov queueing model for duty-cycled nodes with a finite queue capacity. Based on the Markov model, the stationary probability π_{EmptyQ} for a node to have an empty queue can be obtained as a function of the probability p for each node to transmit a data packet in a cycle, i.e., $\pi_{EmptyQ} = f(p)$. On the other hand, to handle the protocol-specific media access rules, we model the probability p for each node to transmit a data packet in a cycle as a function of the probability π_{EmptyQ} of having an empty queue at each node, i.e., $p = g(\pi_{EmptyQ})$. Therefore, solving these two functions, the values of π_{EmptyQ} and p that the investigated protocol is operating on can be obtained, and the throughput, delay and energy consumption of the network can be determined. We show that the analytical throughput, delay and energy consumption of S-MAC and X-MAC using our model match the simulation results, and we also show how our model can be used to optimize the protocol parameters to achieve a desirable performance. Moreover, we suggest how our model can be applied to other synchronized and asynchronous duty-cycled MAC protocols to analyze their performance.

The rest of this paper is organized as follows. Section 2 introduces related work. Section 3 presents our proposed Markov queueing model for duty-cycled nodes. Section 4 provides an overview of analyzing the network throughput, packet delay, and energy consumption using our proposed Markov model. Section 5 and Section 6 elaborate on modeling the queuing dynamics and the stationary probability of packet transmissions for S-MAC and X-MAC, respectively, and we show how the model can be used to optimize protocol parameters. Section 7 discusses the extension of our Markov queuing model and how to apply it to various other duty-cycled MAC protocols. Finally, Section 8 concludes the paper.

II. Related Work

Much work has been done to evaluate the performance of various MAC protocols for wireless sensor networks. Most of the performance evaluations [10][11] are obtained from simulations. However, simulations are usually time consuming and require a large number of runs to obtain statistically significant results. Some other work implemented MAC protocols on motes and obtained their performance using field measurements [12][13]. However, constrained by time, space and available resources, field measurements are oftentimes a case study, from which it is difficult to draw general or quantitative conclusions on the performance of a protocol. Therefore, analytical models are needed to provide insight into the performance of MAC protocols.

Analytical models have been proposed to evaluate the performance of a specific MAC protocol. For example, Bianchi proposed a Markov model to analyze the saturation throughput of IEEE 802.11 [14]. Pollin et al. proposed a Markov model to analyze the performance of slotted IEEE 802.15.4 [15]. Zhang et al. created a Markov model to analyze the throughput, power consumption and packet service delay of S-MAC in a single hop network [16]. However, their model cannot derive the packet queuing delay, which has a significant impact on the packet latency. Zhang et al. proposed an analytical model to evaluate the performance of O-MAC [10]. Although these models can estimate the performance of a specific protocol, they are fundamentally different and cannot be generalized as different protocols have different media access rules. Hence, the application of these models is limited.

Some other work focused on analyzing a specific performance metric, such as delay, or energy consumption, for a specific MAC protocol or a series of similar MAC protocols. For example, Wang et al. analyzed the distribution of the end-to-end delay for CSMA/CA based MAC protocols in wireless sensor networks [17]. Fischione et al. modeled the packet delay in un-slotted IEEE 802.15.4 networks. Wang et al. analyzed the data delivery delay in acoustic sensor networks using queuing theory [18]. Rousselot et al. calculated the lower bound of power consumption for a set of scheduled access and random access MAC protocols for wireless sensor networks [19]. Although some of these works made a general conclusion among certain MAC protocols, their performance evaluations are constrained for only one metric of a protocol's performance. Hence, their models cannot be used to determine the trade-offs in different performance metrics. Luo and et al. created continuous time Markov models and queuing models to analyze the packet loss, delay and power consumption of contention-based MAC protocols with synchronized and asynchronous wake-up patterns [20][21]. However, their models for synchronized wake-up patterns and asynchronous wake-up patterns were different. Moreover, their models obtained the stationary probability of the empty-queue state assuming the number of contending neighbors of each node was known. In fact, the stationary probability of the empty-queue state in return determined the number of contending neighbors of each node in the network. Additionally, their models assume the knowledge of packet transmission rate at each node in a cycle. However, the packet transmission rate at each node is also related to the contention in the network. Hence, the assumption of knowing the number of contending neighbors of each node and the assumption of knowing the packet transmission rate at each node in a cycle are impractical.

Unlike these previous approaches, our model can be used to obtain throughput, delay and energy consumption for both synchronous and asynchronous duty-cycled MAC protocols.

TABLE I

NOTATIONS

Symbol	Quantity	Symbol	Quantity
N	number of nodes in the network	THR	network throughput
T	length of a cycle	THR_p	number of packets received over time
W	contention window size in units of a time slot	PDR	packet delivery ratio
d	duty cycle	D_C	packet contending delay
Q	queue capacity in units of a DATA packet	D_Q	packet queuing delay
S	MAC layer DATA packet size	D	packet delay $D = D_C + D_Q$
P	average energy consumption per second	π	the stationary distribution of the Markov model
E	energy consumption of a node in a cycle	π_0	the stationary probability of the empty-queue state
L	lifetime of each node in the network	π_i	the stationary probability of state i
p_f	probability of transmission failure of a DATA packet		
p_s	probability of successfully transmitting a DATA packet		
p	probability of winning the contention $p = p_s + p_f$		
λ	expected DATA packet arrival rate at the MAC layer		
A_k	probability of k packets arriving in a cycle $A_k = e^{-\lambda T}(\lambda T)^k/k!$		
$A_{\geq k}$	probability of no less than k DATA packets arriving in a cycle $A_{\geq k} = 1 - \sum\limits_{i=0}^{k-1} A_i$		

III. A MARKOV QUEUEING MODEL FOR DUTY-CYCLED NODES

We propose a Markov queueing model for duty-cycled nodes with a fixed cycle length. The model assumes that (1) packets arrive at each node independently, (2) each node can buffer a finite number of DATA packets in a FIFO queue, (3) retransmission is not supported, (4) the channel is ideal (no fading and no capture effect), (5) there is only one transmission opportunity and one DATA packet reception per node per cycle, and (6) every node has a constant probability of transmitting a DATA packet in a cycle regardless of any node's queue length (similar assumptions were made in [14][22], and were verified as good approximations of real scenarios). Table I lists the notations that are used throughout the paper.

The proposed Markov model has a finite number of states, each of which represents a different status of a node, i.e., a different queue length, at the wake-up instant of a cycle. A node may change status cycle by cycle, corresponding to the transition from one state to another in the Markov model. Fig. 1 shows the proposed Markov model with a queue capacity Q. This Markov model has $Q+1$ states, each of which, from left to right, corresponds to 0 packets in the queue, 1 packet in the queue, to Q packets in the queue (full queue). Specifically, when the queue is not empty, a node will attempt to access the media to transmit a DATA packet. A DATA packet is removed from the queue either when it is transmitted successfully, or when it encounters a collision as no retransmission is allowed. A DATA packet is dropped when the queue overflows. Hence, the transition probabilities from one state to another can be described as follows.

$$P_{0,i} = A_i, i = 0..Q-1 \tag{1}$$

$$P_{0,Q} = A_{\geq Q} \tag{2}$$

$$P_{i,i-1} = p \cdot A_0, i = 1..Q \tag{3}$$

$$P_{i,j} = p \cdot A_{j-i+1} + (1-p) \cdot A_{j-i}, i = 1..Q-1, j = i..Q-1 \tag{4}$$

$$P_{i,Q} = p \cdot A_{\geq Q-i+1} + (1-p) \cdot A_{\geq Q-i}, i = 1..Q \tag{5}$$

$$P_{i,j} = 0, i = 2..Q, j = 0..i-2 \tag{6}$$

Equations (1) and (2) describe the fact that all the transitions from the empty-queue state to a non-empty-queue state depend only on new packet arrivals. Equations (3) and (6) describe the fact that a node can only transmit one DATA packet per cycle with a probability p, and the probability of having one packet less in the queue equals the probability of winning the contention times the probability of no packet arrivals in a cycle. Moreover, (4) and (5) describe the fact that the probability of having a non-decreasing queue can be divided into two parts depending on whether the oldest DATA packet in the queue wins the contention (first term) or not (second term). Finally, (2) and (5) show that packets are dropped when the queue overflows.

The proposed Markov model with state space $\tilde{S} = 0, 1, ..., Q$ and transition matrix $\tilde{P}$ has a unique stationary distribution $\pi = (\pi_0, .., \pi_Q)$ since the Markov model is irreducible and aperiodic. Therefore, $\pi_i \geq 0$ for any $s_i \in \tilde{S}$,

$$\sum_{s_i \in \tilde{S}} \pi_i = 1, \pi\tilde{P} = \pi \tag{7}$$

Assuming packet arrival information (λ, A_k, and $A_{\geq}k$) is known, the probability p for each node to win the contention becomes the only variable in the transition matrix $\tilde{P}$. Since π is the unique solution for (7), for any $s_i \in \tilde{S}$, π_i can be represented as a function of p. Specifically,

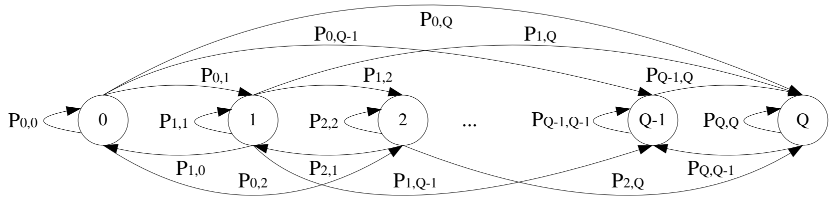


Fig. 1. Markov model for the queuing behavior of duty-cycled nodes without retransmissions.

let function $f(\cdot)$ describe the relationship between π_0 and p, i.e.,

$$\pi_0 = f(p) \tag{8}$$

According to (8), for a given probability p for each node to win the contention, the stationary probability π_0 of the empty-queue state can be obtained using the proposed Markov model.

We propose this Markov model instead of using an M/M/1/Q queuing model because (1) the data packet arrivals at each node in our Markov model can be from distributions other than Poisson, and (2) the service rate at each node (the packet transmission rate at each node in a cycle) is unknown, and it depends on the contention in the network. Meanwhile, our proposed Markov model differs from the Markov models proposed for IEEE 802.11, IEEE 802.16 and IEEE 802.15.4 networks. Our proposed Markov model models the queuing behavior of duty-cycled nodes. Since every state in our Markov model reflects a different queue size at the wake-up instant of a cycle, each transition in our Markov model takes a cycle time. The previous Markov models in [18][14][22], however, model the backoff procedure of a DATA packet. Since the backoff timer of a DATA packet is reset by the duty-cycled MAC protocols at the beginning of each cycle, the previous Markov models cannot handle the duty-cycling behavior of a node.

IV. Protocol-specific Performance Analysis

Our Markov queueing model holds for any duty-cycled nodes with a fixed cycle length. It provides a relationship, as shown in (8), between the stationary probability π_0 of the empty-queue state and the probability p for each node to win the contention in a cycle. However, another relationship between π_0 and p is needed together with (8) to solve for both π_0 and p, and finally using these values to obtain the throughput, delay, and energy consumption. Since (8) obtains π_0 as a function of p, here we show how to use protocol-specific media access rules to obtain p as a function of π_0 as our second relationship between π_0 and p, i.e.,

$$p = g(\pi_0) \tag{9}$$

Given π_0, every node has a probability of π_0 to have an empty queue, and a probability of $1 - \pi_0$ to have a packet to send in a cycle. For a given protocol, the probability p for each node to win the contention can be obtained with the knowledge of π_0 and the media access rules of the protocol. Since different protocols have different media access rules, (9) is protocol-specific. We provide examples of how to obtain (9) for both S-MAC and X-MAC in Sections 5 and 6.

Solving (8) and (9), the stationary probability π_0 of the empty-queue state and the probability p for each node to win the contention can be obtained. Plugging p into (1)-(6), the stationary distribution of the Markov model π can be found. These values enable us to analyze the throughput, delay and energy consumption of the network.

A. *System Model*

We examine the throughput, delay and energy consumption per second per node according to the following system model. A certain number of nodes create a fully-connected network. The nodes are homogeneous in initial energy, power and communication capabilities. Every node has a finite queue to buffer the incoming DATA packets. DATA packet arrivals at different nodes are independent, and they arrive at the nodes with the same distribution. A node randomly selects one of its neighbors as the destination to transmit DATA packets for a certain time. To exclude influences from other than the investigated duty-cycled MAC protocol, routing is not considered.

B. *Throughput Analysis*

Throughput is defined as the amount of data successfully delivered within a unit time. Since the protocols work in a duty-cycled fashion, the throughput can be calculated within a cycle time. Therefore, the throughput of the system can be calculated as follows.

$$THR = N \cdot (1 - \pi_0) \cdot p_s \cdot S/T \tag{10}$$

Since the number of nodes in the network N, the MAC layer DATA packet size S, and the length of a cycle T are known, once π_0 is solved by (8) and (9), the only unknown variable in (10) is the probability p_s for each node to successfully transmit a DATA packet.

When π_0 is known, p_s can be obtained according to the media access rules of the investigated protocol. Similar to the way we obtain p, every node has a packet to send with a probability of $1 - \pi_0$ in a cycle. For a given π_0, the probability p_s for a node to successfully transmit a DATA packet is determined by the number of nodes in the network N and the manner in which nodes compete with each other. The relationship between π_0 and p_s can be described as

$$p_s = h(\pi_0) \tag{11}$$

Plugging (11) into (10), the throughout of the network can be determined. Section 5 and Section 6 provide examples of how to obtain (11) for S-MAC and X-MAC.

Moreover, THR can be used to determine other performance metrics, such as the number of packets received over time THR_p and the packet delivery ratio PDR.

$$THR_p = THR/S = N \cdot (1 - \pi_0) \cdot p_s/T \tag{12}$$

$$PDR = THR/S/N/\lambda = (1 - \pi_0) \cdot p_s/(\lambda \cdot T) \tag{13}$$

where λ is the expected data arrival rate at each node.

C. Delay Analysis

The delay of a DATA packet can be divided into two parts. The first part is the queuing delay D_Q, which is defined as the time interval from when a DATA packet joins the queue at the tail to the DATA packet becoming the head of the queue. The second part is the contending delay D_C, which is defined as the time interval from when the DATA packet is at the head of the queue to when the DATA packet is transmitted and hence removed from the queue. Therefore,

$$D = D_Q + D_C \tag{14}$$

The queuing delay D_Q of a DATA packet is the time that the DATA packet must wait in the queue until all the DATA packets in front of it finish contending for the media. Specifically, a newly joined DATA packet has to wait for a contending delay D_C for each of the DATA packets that are in front of it but behind the head of the queue. However, the newly joined DATA packet may arrive at the queue when the DATA packet at the head of the queue (if the queue is not empty) has already started contending for the media. Hence, for the DATA packet at head of the queue, the newly joined DATA packet has to wait on average for a half of the contending delay D_C of a DATA packet. According to our proposed Markov model, the queuing delay D_Q of a DATA packet can be calculated as a function of the contending delay D_C of a DATA packet.

$$D_Q = D_C \cdot \sum_{i=0}^{Q-1} (max(0, i - 0.5) \cdot \pi_i/(1 - \pi_Q)) \tag{15}$$

The contending delay D_C of a DATA packet can be calculated according to the stationary probability π of our proposed Markov model. A node with a DATA packet to send contends for the media once in a cycle, until the node finally wins the contention. For each contention, the node has a probability of p to win, and a probability of $1-p$ to lose. Given a cycle length of T

$$D_C = T \cdot \sum_{i=0}^{\infty} (i + 1) \cdot p \cdot (1 - p)^i \tag{16}$$

Plugging (15) and (16) into (14), the delay of a DATA packet can be obtained.

D. Energy Consumption Analysis

Since we are considering duty-cycled MAC protocols, the energy consumption per second P of a node can be obtained by calculating the energy consumption E of a node in a cycle divided by the cycle length T, i.e.,

$$P = E/T \tag{17}$$

However, the energy consumption E of a node in a cycle varies as the node plays different roles in the contention. (1) A node could be a sender that successfully transmits a data packet. According to the Markov model, the probability for a node to be a successful sender is $(1 - \pi_0) \cdot p_s$. (2) A node could be a receiver that successfully receives a data packet. Since every sender randomly selects a destination from its neighboring nodes, a node has the same probability of being a successful sender and being a successful receiver, i.e., $(1 - \pi_0) \cdot p_s$. (3) A node could be a sender that encounters a collision. According to the Markov model, the probability for each node to be an unsuccessful sender is $(1 - \pi_0) \cdot p_f$, where $p_f = p - p_s$. (4) A node could be a prospective receiver, but fails to receive the data packet due to a collision. As before, the probability for each node to be an unsuccessful receiver is the same as the probability of being an unsuccessful sender, i.e., $(1 - \pi_0) \cdot p_f$. (5) A node could be idle for its entire active period, as no other nodes attempt to transmit data when the node is awake. The probability of this case can be determined by the media access rules of the protocol. (6) A node could be idle, but it goes to sleep before its active period expires, as some other nodes access the media. The probability of this case can also be determined by the media access rules of the protocol.

To calculate the energy consumption of a node in a cycle, we (1) first calculate the energy consumption of a node in all the above cases. (2) Then, the energy consumption of each case is multiplied by the corresponding probability. (3) The summation of the products obtained in step (2) is the energy consumption of a node in a cycle. The energy consumption per second of a node can then be calculated using (17).

The energy consumption per second P of a node can also be used to estimate the lifetime of the network. Assuming each node in the network has an initial energy E_{init}, the lifetime of the network can be represented as

$$L = E_{init}/P \tag{18}$$

V. Performance Analysis of A Synchronized MAC Protocol - S-MAC

In the following two sections, we provide two examples of how to apply our proposed Markov model and performance analysis methodology to different duty-cycled MAC protocols. Moreover, we show how to optimize protocol parameters using our proposed method to achieve desirable performance. The first example is S-MAC, a synchronized duty-cycled MAC protocol, and the second example is X-MAC, an asynchronous duty-cycled MAC protocol.

S-MAC [3] was the first duty-cycled MAC protocol designed for wireless sensor networks. It is also one of the most popular MAC protocols used for research on and implementation of wireless sensor networks [6][13]. S-MAC operates in a duty-cycled fashion, i.e., sensors sleep and wake up periodically. The active period of a cycle has a fixed length, which is determined by the MAC layer contention window size. The sleeping period of a cycle, instead, could be shorter or longer, depending on the predefined duty cycle, which is the ratio of the active period length to the cycle length. All the nodes in the network have the same cycle length and duty cycle. To improve the communication efficiency, S-MAC synchronizes sensors by exchanging their sleep-awake schedules in SYNC packets, so that every node sleeps and wakes up at the same time. A fixed interval in each active period is reserved for SYNC packet exchange.

S-MAC uses RTS/CTS/DATA/ACK handshaking to guarantee successful unicast transmissions. However, S-MAC has a fixed contention window, as a change in contention window size changes the length of an active period and hence influences the synchronization process. Moreover, when a node fails to win the contention or it encounters an RTS collision, it goes to sleep until the next active period. On the other hand, when a node sends out an RTS successfully, it does not go back to sleep until the transmitted DATA packet is acknowledged.

It is necessary to list all the reasons for DATA packet loss in S-MAC. Assuming ideal channels (i.e., no hidden terminals, capture effect or fading), a DATA packet could be dropped due to (1) overflow of the DATA packet queue, and (2) failure of the associated RTS (RTS collision).

A. Throughput Analysis

Suppose there are N nodes in a fully connected network. When a node has a DATA packet to send, the probability M_k that k out of the other $N-1$ nodes are competing for the media can be described as a function of π_0.

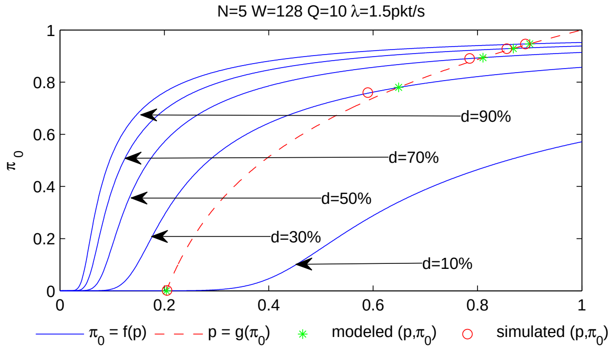


Fig. 2. Determining (p, π_0) using $\pi_0 = f(p)$ and $p = g(\pi_0)$ for S-MAC.

$$M_k(\pi_0) = \binom{N-1}{k} \cdot (1-\pi_0)^k \cdot \pi_0^{N-1-k} = \frac{(N-1)!}{k!(N-1-k)!} \cdot (1-\pi_0)^k \cdot \pi_0^{N-1-k}, k = 0..N-1 \quad (19)$$

In the case that k other nodes are competing for the media, the probability p_k of being the winner (sending out an RTS) and the probability p_{sk} of successfully sending out the DATA packet can be calculated as

$$p_k = \sum_{i=1}^{W} \frac{1}{W} (\frac{W-i+1}{W})^k, k = 0..N-1 \quad (20)$$

$$p_{sk} = \sum_{i=1}^{W} \frac{1}{W} (\frac{W-i}{W})^k, k = 0..N-1 \quad (21)$$

where W is the contention window size. Therefore,

$$p = g(\pi_0) = \sum_{k=0}^{N-1} M_k(\pi_0) \cdot p_k \quad (22)$$

$$p_s = h(\pi_0) = \sum_{k=0}^{N-1} M_k(\pi_0) \cdot p_{sk} \quad (23)$$

Solving (8) and (22), the probability p for each node to win the contention and the stationary probability π_0 of the empty-queue state can be obtained. Plugging π_0 into (23), the probability for each node to successfully transmit a DATA packet, p_s, can be determined. Finally, the throughput of S-MAC can be calculated using (10). Fig. 2 shows an example of (8) and (22). Equation (8) is obtained from the Markov model, and is shown by the solid blue lines for different duty cycles. Equation (22) is obtained from the media access rules of S-MAC, and is shown by the dashed red line. The intersections of these curves, marked by the green asterisks, are the solutions of (8) and (22) for different duty cycles. Each solution corresponds to a specific (p, π_0), which is used to obtain the throughput using (23) and (10). As long as (8) and (22) accurately model the relationships between p and π_0, there is only one solution for (8) and (22) as S-MAC can only operate on a specific (p, π_0) under given duty cycle and network scenario. Validated by NS-2 simulations, our modeled (p, π_0) in Fig. 2 matches the simulated (p, π_0) closely.

B. Delay Analysis

To calculate the delay of S-MAC, the probability p for each node to win the contention and the stationary distribution π of the Markov model need to be determined. p can be obtained by solving (8) and (22) together. Plugging p into (1)-(6), the stationary distribution π of the Markov model can be solved. Then, the delay of S-MAC can be obtained as described in Section 4.2.

C. Energy Consumption Analysis

We define the power for transmitting data, receiving data and sleeping at each node as txp, rxp, and sp, respectively. Note that being idle consumes as much power as receiving data, since a node has to listen to the media. In this paper, $txp = 52.2mW$, $rxp = 59.1mW$, $sp = 0mW$ according to the MICAz datasheet [23], used in both S-MAC and X-MAC energy analysis.

To calculate the energy consumption per second at each node, first it is necessary to recall some S-MAC operations. (1) Each node remains awake during T_{sync} in a cycle to send and receive SYNC packets, (2) each node transmits a SYNC packet every N_{sync} cycles, which is defined as a SYNC period, (3) each node remains awake for a SYNC period to avoid missing any SYNC packets from its neighbors every N_{awake} SYNC periods, hence, a node does not go to sleep in such awake cycles in contrast to going to sleep in the normal cycles, (4) T_{data} is defined as the longest time that a node may need to finish sending an RTS, however, RTS/CTS/DATA/ACK transmissions can last until a new cycle starts, and (5) in normal cycles, a node goes to sleep after transmitting/receiving a DATA packet successfully, or after experiencing an RTS/CTS collision, or after hearing an unintended RTS or CTS from its neighboring nodes.

As a result, a node may consume different amounts of energy in a normal cycle and in an awake cycle. Define E as the average energy consumption in a cycle at each node, E_{normal} as the average energy consumption in a normal cycle, and E_{awake} as the average energy consumption in an awake cycle. We have

$$\begin{aligned} E &= (E_{normal} \cdot N_{sync} \cdot (N_{awake} - 1) + E_{awake} \cdot N_{sync})/(N_{sync} \cdot N_{awake}) \\ &= (E_{normal} \cdot (N_{awake} - 1) + E_{awake})/(N_{sync} \cdot N_{awake}) \end{aligned} \tag{24}$$

Define E_{sync} as the energy consumption during T_{sync}, E_{data} as the energy consumption during data contention and data transmission, E_{sleep} as the energy consumption for sleeping during normal cycles, $E_{nosleep}$ and as the energy consumption for being awake after data contention and data transmission during awake cycles. Therefore,

$$E_{normal} = E_{sync} + E_{data} + E_{sleep} \tag{25}$$

$$E_{awake} = E_{sync} + E_{data} + E_{nosleep} \tag{26}$$

Assume each SYNC, RTS, CTS, DATA, and ACK packet takes t_{SYNC}, t_{RTS}, t_{CTS}, t_{DATA}, and t_{ACK} to transmit, respectively,

$$E_{sync} = ((t_{SYNC} \cdot txp + (T_{sync} - t_{SYNC}) \cdot rxp) + (T_{sync} \cdot rxp \cdot (N_{sync} - 1)))/N_{sync} \tag{27}$$

E_{data}, E_{sleep}, and $E_{nosleep}$ are related with the contention in the network, which determines how long a node needs to be active during random backoff before transmitting data or going to sleep. For a fully-connected network with N nodes, the number of contending nodes in a cycle N_0 has

$$E(N_0) = N \cdot (1 - \pi_0) \tag{28}$$

However, $E(N_0)$ may not be an integer, hence to simplify the calculation, we assume the two cases $N_0 = \lfloor E(N_0) \rfloor$ and $N_0 = \lceil E(N_0) \rceil$ have the highest probabilities and neglect the other possible values of N_0. Define $N_0^- = \lfloor E(N_0) \rfloor$ with a probability of p_-, and $N_0^+ = \lceil E(N_0) \rceil$ with a probability of p_+, we have

$$N_0^- \cdot p_- + N_0^+ \cdot p_+ = E(N_0) \tag{29}$$

$$p_- + p_+ = 1 \tag{30}$$

Plugging (28) into (29) and solving (29) and (30), p_- and p_- can be obtained. Consequently, we can define E_{data-} and E_{data+} as the energy consumption during data contention and data transmission when N_0^- and N_0^+ nodes are contending for the media, respectively.

$$E_{data} = p_- \cdot E_{data-} + p_+ \cdot E_{data+} \tag{31}$$

Similarly,

$$E_{sleep} = p_- \cdot E_{sleep-} + p_+ \cdot E_{sleep+} \tag{32}$$

$$E_{nosleep} = p_- \cdot E_{nosleep-} + p_+ \cdot E_{nosleep+} \tag{33}$$

When $N_0^- = 0$, no node is contending for the media in this case. Hence,

$$E_{data-} = T_{data} \cdot rxp \tag{34}$$

$$E_{sleep-} = (T - T_{sync} - T_{data}) \cdot sp \tag{35}$$

$$E_{nosleep-} = (T - T_{sync} - T_{data}) \cdot rxp \tag{36}$$

When $N_0^- > 0$ and hence $N_0^+ > 0$, a node has a probability $(1-\pi_0) \cdot p_s$ of being the sender of a successful DATA transmission, implying it also has the same probability of being the destination of a successful DATA transmission. Similarly, a node has a probability $(1 - \pi_0) \cdot p_f$ of being the sender of an unsuccessful DATA transmission, implying it also has the same probability of being the destination of an unsuccessful DATA transmission. Assume (1) each packet has a propagation delay of D_{prop}, (2) W_{s-} and W_{s+} are the average backoff windows of the winner that successfully transmits a DATA packet in the case of N_0^- and N_0^+, respectively, (3) W_{c-} and W_{c+} are the average backoff windows of the winner that experiences a collision in the case of N_0^- and N_0^+, respectively, and (4) W_{t-} and W_{t+} are the average backoff windows of a winner of the contention in the case of N_0^- and N_0^+, respectively. Since the calculations for both N_0^- and N_0^+ are identical, to reduce the redundancy, only the calculations of E_{data-}, E_{sleep-}, and $E_{nosleep-}$ are shown as follows. To calculate E_{data+}, E_{sleep+}, and $E_{nosleep+}$, simply replace the subscript/superscript ”−” with ”+” in equations (37)-(42).

$$\begin{aligned}
&E_{data-} = \\
&(1 - \pi_0) \cdot p_s \cdot ((t_{RTS} + t_{DATA}) \cdot txp + (t_{CTS} + t_{ACK}) \cdot rxp + 4 \cdot D_{prop} \cdot rxp + W_{s-} \cdot rxp \\
&+(1 - \pi_0) \cdot p_s \cdot ((t_{RTS} + t_{DATA}) \cdot rxp + (t_{CTS} + t_{ACK}) \cdot txp + 3 \cdot D_{prop} \cdot rxp + W_{s-} \cdot rxp \\
&+(1 - \pi_0) \cdot p_f \cdot (t_{RTS} \cdot txp + t_{CTS} \cdot rxp + 2 \cdot D_{prop} \cdot rxp + W_{c-} \cdot rxp) \\
&+(1 - \pi_0) \cdot p_f \cdot (t_{RTS} \cdot rxp + t_{CTS} \cdot txp + D_{prop} \cdot rxp + W_{c-} \cdot rxp) \\
&+(1 - 2 \cdot (1 - \pi_0) \cdot (p_s + p_f)) \cdot (t_{RTS} \cdot rxp + W_{t-} \cdot rxp)
\end{aligned} \tag{37}$$

$$\begin{aligned}
&E_{sleep-} = \\
&(1 - \pi_0) \cdot p_s \cdot (T - T_{sync} - W_{s-} - t_{RTS} - t_{CTS} - t_{DATA} - t_{ACK} - 4 \cdot D_{prop}) \cdot sp \\
&+(1 - \pi_0) \cdot p_s \cdot (T - T_{sync} - W_{s-} - t_{RTS} - t_{CTS} - t_{DATA} - t_{ACK} - 3 \cdot D_{prop}) \cdot sp \\
&+(1 - \pi_0) \cdot p_f \cdot (T - T_{sync} - W_{c-} - t_{RTS} - t_{CTS} - 2 \cdot D_{prop}) \cdot sp
\end{aligned}$$

$$+(1-\pi_0)\cdot p_f\cdot(T-T_{sync}-W_{c-}-t_{RTS}-t_{CTS}-D_{prop})\cdot sp$$

$$+(1-2\cdot(1-\pi_0)\cdot(p_s+p_f))\cdot(T-T_{sync}-t_{RTS}-T_{t-})\cdot sp \tag{38}$$

$$\begin{aligned}
&E_{nosleep-}=\\
&(1-\pi_0)\cdot p_s\cdot(T-T_{sync}-W_{s-}-t_{RTS}-t_{CTS}-t_{DATA}-t_{ACK}-4\cdot D_{prop})\cdot rxp\\
&+(1-\pi_0)\cdot p_s\cdot(T-T_{sync}-W_{s-}-t_{RTS}-t_{CTS}-t_{DATA}-t_{ACK}-3\cdot D_{prop})\cdot rxp\\
&+(1-\pi_0)\cdot p_f\cdot(T-T_{sync}-W_{c-}-t_{RTS}-t_{CTS}-2\cdot D_{prop})\cdot rxp\\
&+(1-\pi_0)\cdot p_f\cdot(T-T_{sync}-W_{c-}-t_{RTS}-t_{CTS}-D_{prop})\cdot rxp\\
&+(1-2\cdot(1-\pi_0)\cdot(p_s+p_f))\cdot(T-T_{sync}-t_{RTS}-T_{t-})\cdot rxp
\end{aligned} \tag{39}$$

However, to calculate (37), (38), and (39), the average backoff window W_{s-} of the contention winner that successfully transmits a DATA packet, the average backoff window W_{c-} of the contention winner that experiences a collision, and the average backoff window W_{t-} of the contention winner need to be obtained.

First, we calculate W_{s-}, the average backoff window of the contention winner that successfully transmits a DATA packet. For each possible backoff window from 0 to $W-1$, a node wins the contention and successfully transmits a DATA packet if all the other $N_0^- - 1$ nodes have larger backoff windows. Therefore,

$$W_{s-}=\sum_{i=0}^{W-1} i\cdot(\binom{N_0^-}{1}\cdot\frac{1}{W}\cdot(\frac{W-i-1}{W})^{N_0^- -1}/\sum_{j=0}^{W-1}(\binom{N_0^-}{1}\cdot\frac{1}{W}\cdot(\frac{W-j-1}{W})^{N_0^- -1})) \tag{40}$$

Similarly, we calculate W_{c-}, the average backoff window of the contention winner that experiences a collision. For each possible backoff window from 0 to $W-1$, a node wins the contention but has a collision if (1) all the other $N_0^- - 1$ nodes have equal or larger backoff windows, and (2) at least one contending neighbor has the same backoff window. Therefore,

$$W_{c-}=\sum_{i=0}^{W-1} i\cdot\frac{(\frac{W-i}{W})^{N_0^-}-(\frac{W-i-1}{W})^{N_0^-}-\binom{N_0^-}{1}\cdot\frac{1}{W}\cdot(\frac{W-i-1}{W})^{N_0^- -1}}{\sum\limits_{j=0}^{W-1}((\frac{W-i}{W})^{N_0^-}-(\frac{W-i-1}{W})^{N_0^-}-\binom{N_0^-}{1}\cdot\frac{1}{W}\cdot(\frac{W-i-1}{W})^{N_0^- -1})} \tag{41}$$

Finally, we calculate W_{t-}, the average backoff window of the contention winner. For each possible backoff window from 0 to $W-1$, a node wins the contention if all the other $N_0^- - 1$ nodes have equal or larger backoff windows. Therefore,

$$W_{t-} = \sum_{i=0}^{W-1} i \cdot \frac{(\frac{W-i}{W})^{N_0^-} - (\frac{W-i-1}{W})^{N_0^-}}{\sum_{j=0}^{W-1}((\frac{W-j}{W})^{N_0^-} - (\frac{W-j-1}{W})^{N_0^-})} \tag{42}$$

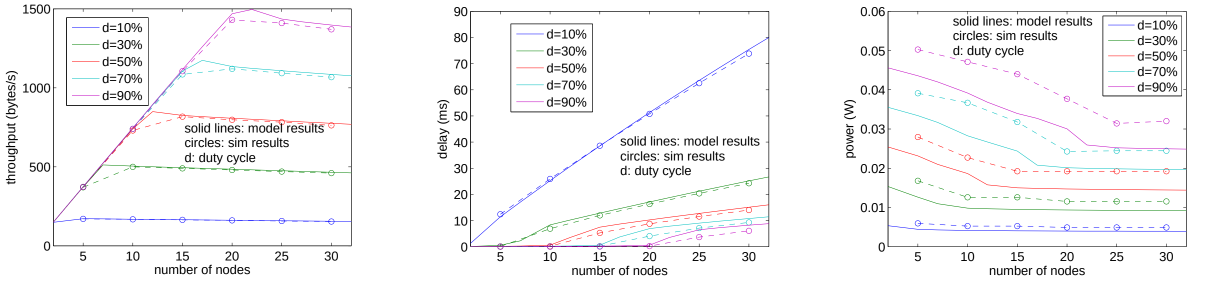
Plugging (40), (41) and (42) into (37), (38) and (39), E_{data-}, E_{sleep-}, and $E_{nosleep-}$ are obtained. Replacing subscript/superscript ”$-$” with ”$+$” in (37)-(42), E_{data+}, E_{sleep+}, and $E_{nosleep+}$ can be obtained. According to the value of N_0^- and N_0^+, substituting E_{data-}, E_{sleep-}, $E_{nosleep-}$, E_{data+}, E_{sleep+}, and $E_{nosleep+}$ into (31), (32), and (33), E_{data}, E_{sleep}, and $E_{nosleep}$ can be obtained. Calculating E_{sync} through (27), and substituting E_{sync}, E_{data}, E_{sleep}, and $E_{nosleep}$ into (25) and (26), and finally (24), the average energy consumption per node in a cycle E is obtained. Hence, the energy consumption per second per node P can be obtained using (17).

D. Model Validations

To validate our proposed Markov model, we run the model under various S-MAC configurations and data arrival rates, and compare the estimates of throughput, delay, and energy consumption with simulation results using NS-2. In the simulation, (1) the network is fully-connected, (2) every 100s one of a node's neighbors is randomly selected as the destination of the packets that arrive in the following 100s, (3) the simulation time is 2000s, and (4) all the results (throughput, delay, and energy consumption) are averaged over 50 runs.

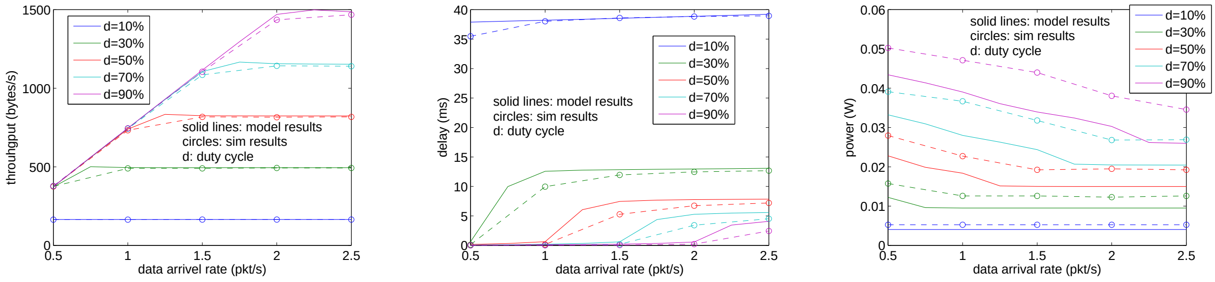
In all the simulations, the basic set-up is a fully-connected network with $N = 15$ nodes, a contention window size W of 128, a data arrival rate λ at each node of 1.5 packets per second, and a queue capacity Q at each node of 10. For each set of simulations, we vary one of these parameters and investigate the throughput, delay and average energy consumption per second of S-MAC. Fig. 3 shows the performance results under varying node density from 5 nodes in the networks to 30 nodes in the network. Fig. 4 shows the performance results under varying data arrival rate λ from 0.5 packets per second to 2.5 packets per second at each node. Fig. 5 shows the performance results under varying contention window size W from 64 to 512. Fig. 3, Fig. 4 and Fig. 5 also show the 95% confidence intervals of the simulation results. From these figures, we can see that our analytical results using the Markov model match the simulation results well for throughput and delay and approximate the simulation results for energy consumption.

In Fig. 3a, the throughput of S-MAC first increases linearly as the number of nodes in the network increases, and then it stops increasing and decreases slightly as the node density further



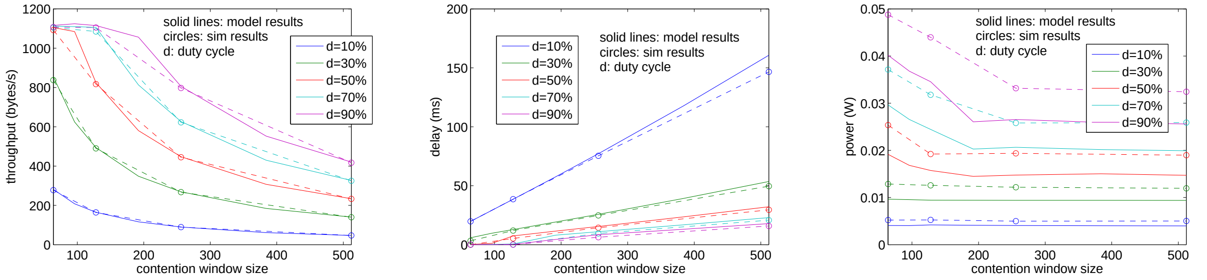
(a) Throughput. (b) Delay. (c) Energy consumption/s/node.

Fig. 3. S-MAC performance with different numbers of nodes in the network. (95% confidence interval is shown)



(a) Throughput. (b) Delay. (c) Energy consumption/s/node.

Fig. 4. S-MAC performance with different data arrival rates. (95% confidence interval is shown)



(a) Throughput. (b) Delay. (c) Energy consumption/s/node.

Fig. 5. S-MAC performance with different contention window sizes. (95% confidence interval is shown)

increases. Similarly, in Fig. 4a, the throughput of S-MAC first increases linearly as the data arrival rate increases, however, the throughput stops increasing and remains constant as the data arrival rate further increases. In Fig. 5a, the throughput of S-MAC first remains constant (S-MAC with 70% and 90% duty cycles), and then decreases as the contention window size increases. In all these figures, there is a change in the trend of throughput. Before the turning point of the throughput curves, S-MAC can deliver all the DATA packets as soon as they arrive in the network. Hence, the throughput increases linearly as the number of nodes increases, or the data arrival rates increases, and the throughput of S-MAC remains constant as the contention window size increases. However, after the turning point of the throughput curves, S-MAC saturates, i.e., S-MAC reaches its delivering limit and can no longer deliver all the incoming DATA packets, and thus DATA packets are backlogged in the queue until the queue overflows. As the node density keeps increasing, the contention in the network further increases. Hence, the probability of successfully transmitting a packet decreases, and more packets are finally dropped due to collisions and queue overflow. Consequently, the throughput decreases as the node density increases. Different from increasing the node density in the network, increasing the data arrival rate after S-MAC saturates does not increase the contention in the network since every node always has packets to send in each cycle. Therefore, more arriving packets are dropped due to queue overflow instead of collisions, and hence the throughput of S-MAC does not decrease. On the other hand, as the contention window size increases, the cycle length of S-MAC increases. Since S-MAC delivers at most one DATA packet per cycle, longer cycle length leads to lower throughput. Therefore, as the cycle length increases, the throughput of S-MAC decreases.

In Fig. 3b, the delay of S-MAC first remains nearly zero, and then increases as the node density increases or the contention window size increases. Before S-MAC saturates, S-MAC can deliver incoming packets as soon as they arrive at the network, hence few packets are accumulated in the queue, and as a result, the packet delay is very small. However, as the node density increases, (1) the contention in the network increases, hence a packet has a longer contending delay before its associated RTS is sent, and (2) when S-MAC saturates, packets have a longer queuing delay as the average queue length increases. Therefore, the delay of S-MAC increases as the node density increases. In Fig. 4b, the delay of S-MAC jumps from very low to a certain value as the data arrival rates increases. Again, S-MAC has very small delay when it is far from saturation due to the very low data arrival rates. As the data arrival rate increases, the packet delay increases because the average queue length at each node increases dramatically. However, the average queue length cannot be longer than the queue capacity. Moreover, increasing the data arrival rate does not intensify the contention in the network after S-MAC saturates. Therefore, neither the contention delay nor the queuing delay of a packet increases. Hence, the packet delay remains constant after S-MAC saturates. For varying contention window size (Fig. 5b), after S-MAC saturates, all the nodes in the network have a packet to send in every cycle, and therefore the contention in the network (specifically the probability p for each node to transmit a DATA packet and the stationary distribution π of the Markov model) remain almost constant as the contention window size increases. Since p and π are not sensitive to the contention window size after S-MAC saturates, according to (15) and (16) both contending delay and queuing delay increase linearly as the cycle length increases. Therefore, after S-MAC saturates the delay of S-MAC increases linearly as the contention window size increases.

In Fig. 3c, Fig. 4c, and Fig. 5c, the average energy consumption per node per second of S-MAC decreases as the number of nodes in the network increases, or the data arrival rate increases, or the contention window size increases. When there is little traffic in the network that has an extremely low node density or data arrival rate or contention window size, few nodes are contending for the media and all nodes are idle-listening to the channel during T_{data} in every normal cycle. As the node density or the data arrival rates increase, more nodes have packets to send in a cycle, and hence more nodes hear an unintended RTS and go to sleep before T_{data} expires. Meanwhile, as more nodes are contending for the media, the average backoff window of a contention winner decreases, which means nodes tend to hear an RTS earlier and go to sleep for a longer time in each cycle. Hence, on average, the energy consumption per second of each node decreases as the node density or the data arrival rate increases. When S-MAC saturates, a node can either be a contention winner or go to sleep after hearing an unintended RTS, but the chance of being awake and hearing nothing during T_{data} is small. For varying node density (Fig. 3c), as the node density increases, the average backoff window of a contention winner decreases slowly, which means less energy can be saved by hearing an earlier RTS. Therefore, the average energy consumption per second at each node decreases much slower than before. However, for varying data arrival rate (Fig. 4c), when S-MAC saturates, every node has a packet to send during each cycle, hence the average backoff window of a contention winner remains constant, and therefore, the average energy consumption per second cannot be decreased any more. For varying contention window size (Fig. 5c), a longer contention window means that unintended nodes have to to be awake for a longer time until the first RTS is sent. So, the energy consumption of a node per cycle increases as the cycle length increases. However, a longer contention window size also means a longer cycle length. In our experiment, the average energy consumption per second decreases slowly when S-MAC saturates.

Fig. 3, Fig. 4, and Fig. 5 also shows that for a given node density, S-MAC with a higher duty cycle has throughput and packet delay no worse than S-MAC with a lower duty cycle. However, S-MAC with a higher duty cycle always has higher energy consumption per second than S-MAC with a lower duty cycle, no matter whether S-MAC saturates or not.

E. S-MAC Optimizations

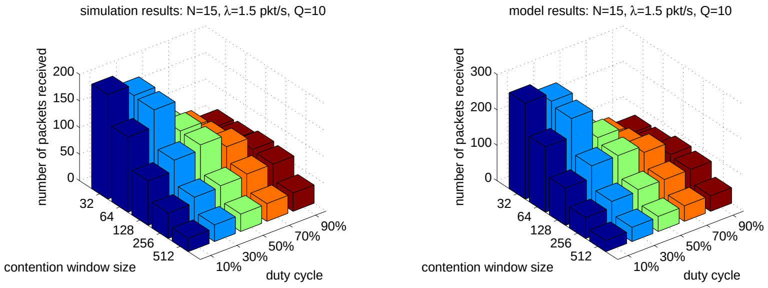
Our proposed Markov model can be used to estimate the throughput, delay, and energy consumption of S-MAC under different S-MAC configurations, network conditions and data arrival rates. Since S-MAC is the first, and the most basic, duty-cycled MAC protocol for wireless sensor networks, it is oftentimes used as a baseline in performance comparisons with newly proposed duty-cycled MAC protocols. Hence, future research on duty-cycled MAC protocols can benefit from the convenience and flexibility provided by our Markov model for S-MAC to avoid massive simulations.

Moreover, our proposed Markov model and performance analysis can be used to optimize the protocol parameters in order to achieve desirable performance, which may have requirements on throughput, delay, and energy consumption simultaneously. In this section, we show an example of optimizing the duty cycle and contention window size of S-MAC, in order to receive the most DATA packets in the lifetime of a network. The number of packets received at each node Pkt_{recvd} can be obtained by

$$Pkt_{recvd} = THR/S/N \cdot L = THR_p/N \cdot L = PDR \cdot (\lambda \cdot L) \tag{43}$$

where S is the DATA packet size, N is the number of nodes in the network, λ is the expected data arrival rate at each node, and L is the lifetime of each node in the network. Equation (43) can be obtained by (10) and (18) using our proposed Markov model and performance analysis.

Fig. 6 shows the number of packets a node can receive during its lifetime under different duty cycles and contention window sizes, using our model and simulations. Our model provides a good estimation of the optimal duty cycle and contention window size. In our experiment, there are 15 nodes in a fully-connected network, the data arrival rate is 1.5 packets per second at each node, and the queue capacity is 10. The available duty cycles are 10%, 30%, 50%, 70%,



(a) Simulation Results.

(b) Analytical Results.

Fig. 6. S-MAC performance optimization under different contention window sizes and duty cycles.

and 90%, and the available contention window sizes are 32, 64, 128, 256, and 512. In order to receive the most DATA packets at each node, the optimal contention window is 32 and the optimal duty cycle is 10%.

VI. Performance Analysis of An Asynchronous MAC Protocol - X-MAC

In this section, we apply our proposed Markov queuing model and performance analysis to X-MAC, an asynchronous duty-cycled MAC protocol for wireless sensor networks. X-MAC avoids synchronization overhead, and hence it has higher energy-efficiency than synchronized MAC protocols such as S-MAC. Additionally, X-MAC uses a series of short preamble packets instead of an extended preamble, like B-MAC [4]. The short preamble packets carry the address information of the destination node. As a result, non-destination nodes can go to sleep as soon as they hear the first short preamble instead of remaining awake until the extended preamble ends. Moreover, the destination node can reply with an ACK in between two successive short preambles to stop the preamble and start the data transfer. Therefore, this strobbed preamble approach saves energy and greatly reduces latency. Furthermore, X-MAC has a fixed preamble size, and hence it can be readily adapted to the packetized radios that are emerging as the standard in today's sensor motes [24].

In X-MAC, every node wakes up periodically to send and receive packets. The interval between two successive wake-ups is a cycle. X-MAC has a fixed cycle length for every node, yet each node starts its duty cycle with an arbitrary offset. As a result, when a sender wakes up to send a packet, the receiver may still be sleeping. Hence, the sender, from the time it wakes up, starts sending short preamble packets with the receiver's address information. In between two successive preamble packets, the sender pauses to listen to the media. At some point, the receiver wakes up and hears the preamble. The receiver sends an ACK back to the sender during the pause between the two preambles. Note that the pause is shorter than the time that a node needs to detect an ongoing transmission, hence only the receiver can access the channel during the pause, while other nodes cannot interfere with the communication between the sender and the receiver [24]. When the sender receives the ACK, it starts sending the DATA packet as the receiver is ready to receive. If a node wakes up without any packet to send, it goes to sleep if (1) the node hears an intended preamble packet, or (2) the node does not hear any transmission for a fixed amount of time, which is defined as the active time of a node, T_{active}, in a cycle.

Since X-MAC is asynchronous, for each DATA packet that is successfully delivered, the average communication time for the sender is $T/2$ plus the time to transmit a DATA packet t_{DATA} [24]. X-MAC also has collisions. When more than one sender wake up and start sending their preambles at the same time, all the other nodes, including the receivers, cannot determine the destination address information in the preambles. In this case, the senders will not stop sending preambles until their next wake-up time. Hence, for each colliding DATA packet, the average communication time for the sender is T [24]. For simplicity, in this paper we analyze the throughput of slotted X-MAC. Hence, T, T_{active}, t_{DATA}, and other timing parameters in the following analysis are in the unit of a time slot, τ.

A. Throughput Analysis

Define p_f to be the probability of a collision when a node transmits a DATA packet in a cycle.

$$p = p_s + p_f \tag{44}$$

We define (1) $empty$ and $\overline{empty}$ to be the status of the queue when a node wakes up. Hence the probability of having the queue "$empty$" is π_0 according to our proposed Markov model. (2) $free$ and $busy$ to be the status of the channel, and (3) A to be the event that a node becomes the only one winner of the contention, and B to be the event that a node becomes one of the multiple winners in the contention (implying a collision). According to the Markov model

$$p_s = Pr(A, free|\overline{empty}) = Pr(A|free, \overline{empty}) \cdot Pr(free|\overline{empty}) \tag{45}$$

$$p_f = Pr(B, free|\overline{empty}) = Pr(B|free, \overline{empty}) \cdot Pr(free|\overline{empty}) \tag{46}$$

We solve for $Pr(A|free, \overline{empty})$ and $Pr(B|free, \overline{empty})$, and then determine $Pr(free|\overline{empty})$.

Given a node has packets in its queue, and the channel is free when it wakes up, the node can successfully transmit a DATA packet if (1) no other nodes in the network wake up at the same time, or (2) some nodes wake up at the same time, but they have no packets to send. Hence,

$$Pr(A|free, \overline{empty}) = \sum_{t=1}^{T} \frac{1}{T} (\sum_{i=0}^{N-1} \binom{N-1}{i} (\frac{1}{T})^i {\pi_0}^i (\frac{T-1}{T})^{N-1-i}) \tag{47}$$

Similarly, given a node has packets to send in the queue, and the channel is free when it wakes up, the node has a collision when transmitting a DATA packet if at least one other node in the network wakes up at the same time and has packets to send. Hence,

$$Pr(B|free,\overline{empty}) = \sum_{t=1}^{T}\frac{1}{T}(\sum_{i=1}^{N-1}\binom{N-1}{i}(\frac{1}{T})^i(\sum_{j=1}^{i}(1-\pi_0)^j{\pi_0}^{i-j})(\frac{T-1}{T})^{N-1-i}) \tag{48}$$

$Pr(free|\overline{empty})$ is the probability of a free channel when a node wakes up with packets to send in its queue. Since in X-MAC every node wakes up and sends packets with an arbitrary offset to other nodes, the channel experiences the same probability to be $free$ or $busy$ in every time slot. Hence, when a single node wakes up, no matter whether its queue is empty or not, the node sees the channel with the similar probability of being $free$ or $busy$. Therefore,

$$Pr(free) \approx Pr(free|\overline{empty}) \tag{49}$$

This approximation is validated by comprehensive simulations using Matlab. Consequently, the problem of determining $Pr(free|\overline{empty})$ thus becomes the problem of determining $Pr(free)$.

The probability $Pr(free)$ of a free channel can be obtained if we know: (1) the average length E_{free} of a free channel between two transmissions over the media, and (2) the average length E_{busy} of a busy channel between the two chunks of a free channel.

$$Pr(free) = \frac{E_{free}}{E_{free}+E_{busy}} \tag{50}$$

To calculate E_{free}, consider the time instant when a transmission ends and a chunk of free channel begins (note that the length of a specific chunk of free channel could be zero). From that time instant, the channel could be free for a certain number of cycles, say n cycles, until in the $n+1^{st}$ cycle, some node(s) start to transmit at the t^{th} slot. The length of this chunk of free channel is $n \cdot T + t$, and the probability of this event $P_{free}(n,t)$ can be obtained as

$$P_{free}(n,t) = \pi_0^{Nn}\cdot\sum_{i=0}^{N-1}\sum_{j=1}^{N-i}\sum_{k=1}^{j}\binom{N}{i}(\frac{t}{T})^i{\pi_0}^i\binom{N-i}{j}(\frac{1}{T})^j\binom{j}{k}(1-\pi_0)^k{\pi_0}^{j-k}(\frac{T-t-1}{T})^{N-i-j} \tag{51}$$

Therefore, the average length of a free channel between two transmissions over the media

$$E_{free} = \sum_{n=0}^{\infty}\sum_{t=0}^{T-1}(nT+t)\cdot P_{free}(n,t) \tag{52}$$

Similarly, if the channel is free for n cycles and t slots, the probability that a transmission is successful is $P_{busy}^{suc}(n,t)$, and the probability that a collision occurs is $P_{busy}^{col}(n,t)$.

$$P_{busy}^{suc}(n,t) = \pi_0^{Nn}\cdot\sum_{i=0}^{N-1}\sum_{j=1}^{N-i}\binom{N}{i}(\frac{t}{T})^i{\pi_0}^i\binom{N-i}{j}(\frac{1}{T})^j\binom{j}{1}(1-\pi_0){\pi_0}^{j-1}(\frac{T-t-1}{T})^{N-i-j} \tag{53}$$

$$P_{busy}^{col}(n,t) = \pi_0^{Nn} \cdot \sum_{i=0}^{N-2} \sum_{j=2}^{N-i} \sum_{k=2}^{j} \binom{N}{i} (\frac{t}{T})^i {\pi_0}^i \binom{N-i}{j} (\frac{1}{T})^j \binom{j}{k} (1-\pi_0)^k {\pi_0}^{j-k} (\frac{T-t-1}{T})^{N-i-j} \tag{54}$$

The average length E_{busy} of a busy channel between the two chunks of a free channel can be calculated as the average length of a successful transmission $T/2 + t_{DATA}$ times the probability of a successful transmission, plus the length of a colliding transmission T times the probability of a collision. Hence,

$$E_{busy} = \sum_{n=0}^{\infty} \sum_{t=0}^{T-1} ((T/2 + t_{DATA}) \cdot P_{busy}^{suc}(n,t) + T \cdot P_{busy}^{col}(n,t)) \tag{55}$$

Plugging (52) and (55) into (50), we obtain $Pr(free)$. Then plugging (50) into (49), we obtain $Pr(free|\overline{empty})$. Plugging (47) and (49) into (45), and (48) and (49) into (46), the probability p_s for each node to successfully transmit a DATA packet and the probability p_f for each node to encounter a collision are solved as a function of the stationary probability π_0 of the empty-queue state. According to (44), the probability p for each node to transmit a DATA packet can also be obtained as a function of π_0. Let function $g(\cdot)$ describe the relationship between p and π_0,

$$p = g(\pi_0) \tag{56}$$

Solving (8) and (56), the actual p and π_0 under which X-MAC is operating for a given scenario can be determined. Plugging in the actual π_0 to (45), the probability p_s for each node to successfully deliver a DATA packet can be obtained. The throughput of X-MAC can therefore be solved as follows according to (10) with a slight modification due to the time unit.

$$THR = N \cdot (1 - \pi_0) \cdot p_s \cdot S/(T \cdot \tau) \tag{57}$$

B. Delay Analysis

According to (14)-(16), to calculate the delay of X-MAC, the probability p for each node to win the contention and the stationary distribution π of the Markov model need to be determined. p can be obtained by solving (8) and (56) together. Plugging the obtained p into (1)-(6), the stationary distribution π of the Markov model can be solved. Therefore, the delay of X-MAC can be obtained as described in Section 4.2 with a slight modification to (16).

$$D_C = T \cdot \tau \cdot \sum_{i=0}^{\infty} (i+1) \cdot p \cdot (1-p)^i \tag{58}$$

C. Energy Consumption Analysis

Define an active period of a node in a cycle as T_{active} time units, and assume a preamble packet, a DATA packet, and an ACK packet take t_{pre}, t_{DATA}, and t_{ACK} time units to transmit, respectively.

For a fully-connected network with N nodes, a node has a probability $(1-\pi_0)\cdot p_s$ of being the sender of a successful DATA transmission, which takes on average $T/2+t_{DATA}$ to finish. During the $T/2+t_{DATA}$ communication period, t_{DATA} is used to send the DATA packet, while the rest $T/2$ is used to periodically send preamble packets and listen to the ACK packets in between every two successive preamble packets. Hence, $T/2\cdot(t_{pre}/(t_{pre}+t_{ACK}))$ is used to send preambles packets and $T/2\cdot(t_{ACK}/(t_{pre}+t_{ACK}))$ is used to listen to the media. Therefore, the energy consumed in this case E_1 can be obtained as

$$E_1=(1-\pi_0)\cdot p_s\cdot\tau\cdot(T/2\cdot(t_{pre}/(t_{pre}+t_{ACK}))\cdot txp+T/2\cdot(t_{ACK}/(t_{pre}+t_{ACK}))\cdot rxp+t_{DATA}\cdot rxp) \tag{59}$$

Meanwhile, a node has a probability $(1-\pi_0)\cdot p_s$ of being the destination of a successful DATA transmission. In this case, the node has to receive a complete preamble packet for t_{pre}, send an ACK packet for t_{ACK}, and receive the DATA packet for t_{DATA}. However, the receiving node may wake up when the transmitting node is half way through sending a preamble packet or listening for an ACK packet. As a result, the receiving node can only catch the next preamble packet sent by the transmitting node. Therefore, on average the receiving node listens to the media for $(t_{pre}+t_{ACK})/2$ before it receives a complete preamble packet. The energy consumed in this case E_2 can be obtained as

$$E_2=(1-\pi_0)\cdot p_s\cdot\tau\cdot((t_{pre}+t_{ACK})/2\cdot rxp+t_{pre}\cdot rxp+t_{ACK}\cdot txp+t_{DATA}\cdot rxp) \tag{60}$$

Similarly, a node has a probability $(1-\pi_0)\cdot p_f$ of being the sender of an unsuccessful DATA transmission, implying at least one other node starts to send preambles at the same time. In this case, no preamble packets can be correctly received, and hence the senders keep transmitting preamble packets and listen to the media during the entire cycle T. Therefore, $T\cdot(t_{pre}/(t_{pre}+t_{ACK}))$ is used to send preamble packets and $T\cdot(t_{ACK}/(t_{pre}+t_{ACK}))$ is used to listen to the media in between two successive preamble packets. The energy consumed in this case E_3 can be obtained as

$$E_3=(1-\pi_0)\cdot p_f\cdot\tau\cdot(T\cdot(t_{pre}/(t_{pre}+t_{ACK}))\cdot txp+T\cdot(t_{ACK}/(t_{pre}+t_{ACK}))\cdot rxp) \tag{61}$$

A node also has a probability $(1-\pi_0)\cdot p_f$ of being the destination of an unsuccessful DATA transmission. If the node wakes up when the colliding preamble packets are half way through transmission or when the colliding senders are listening to the media between two successive preamble packets, the node cannot detect the collision until it hears the next colliding preamble packets. Hence, on average the node keeps awake for $(t_{pre}+t_{ACK})/2+t_{pre}$ before going to sleep. The energy consumed in this case E_4 can be obtained as

$$E_4 = (1-\pi_0)\cdot p_f\cdot\tau\cdot((t_{pre}+t_{ACK})/2\cdot rxp + t_{pre}\cdot rxp) \tag{62}$$

At last, there is a probability of $1-2\cdot(1-\pi_0)\cdot(p_s+p_f)$ that a node is not involved in any data transmission in a cycle. The node goes to sleep if (1) it hears a complete preamble packet, or (2) T_{awake} expires. In this case, the energy consumption E_5 can be obtained as

$$\begin{aligned}&E_5 = (1-2\cdot(1-\pi_0)\cdot(p_s+p_f))\cdot\tau\cdot rxp\,\cdot\\&(\sum_{t=0}^{T_{active}-1} P_{free}(0,t)\cdot(t+(t_{pre}+t_{ACK})/2+t_{pre})\\&+(\sum_{t=T_{active}}^{T-1} P_{free}(0,t)+\sum_{n=1}^{\infty}\sum_{t=0}^{T-1} P_{free}(n,t))\cdot T_{active})\end{aligned} \tag{63}$$

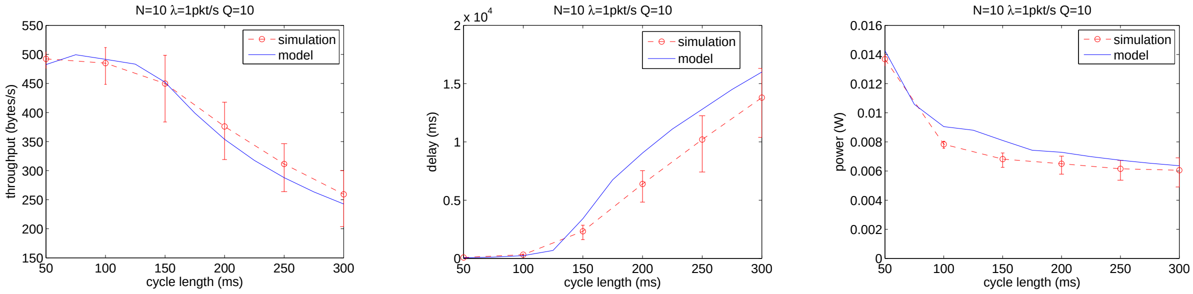
Overall, the energy consumption E of a node in a cycle can be obtained as

$$E = E_1+E_2+E_3+E_4+E_5 \tag{64}$$

The energy consumption per second P of an X-MAC node can be obtained according to (17).

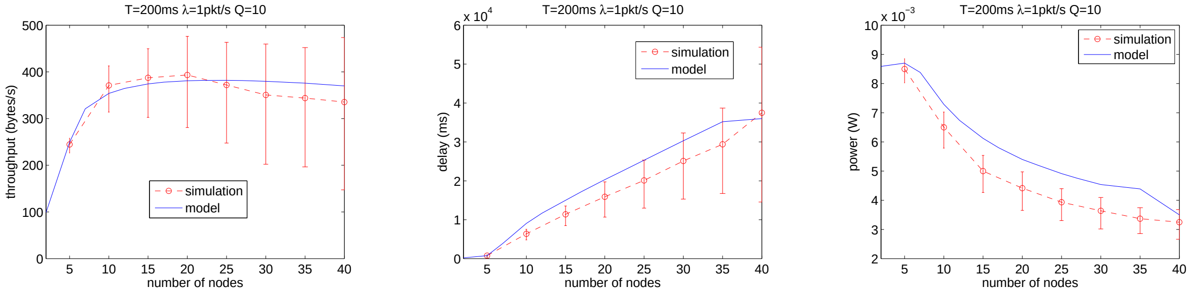
D. Model Validations

To validate our proposed Markov model, we run the model under various X-MAC configurations and data arrival rates, and compare the estimations of throughput, delay, and average energy consumption per second with the simulation results using Matlab. In the simulation, (1) the network is fully-connected, (2) each node randomly selects one of its neighbors as the destination for every packet, (3) the simulation time is 1000s, (4) all the results (throughput, delay, and energy consumption) are averaged over 50 simulations, and (5) the cycle length T is 200ms. A time unit is 1ms. The active time in each cycle T_{active} is 15. The time used to transmit a preamble packet t_{pre} is 3. The time used to transmit an ACK packet t_{ACK} is 1. The time used to transmit a DATA packet t_{DATA} is 5.



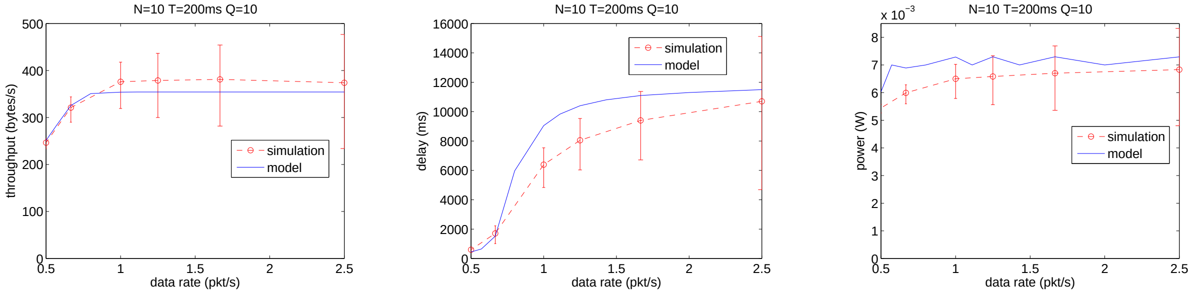
(a) Throughput. (b) Delay. (c) Energy consumption/s/node.

Fig. 7. X-MAC performance with different cycle lengths. (95% confidence interval is shown)



(a) Throughput. (b) Delay. (c) Energy consumption/s/node.

Fig. 8. X-MAC performance with different numbers of nodes in the network. (95% confidence interval is shown)



(a) Throughput. (b) Delay. (c) Energy consumption/s/node.

Fig. 9. X-MAC performance with different data arrival rates. (95% confidence interval is shown)

In all the simulations, the basic set-up is a fully-connected network with $N = 10$ nodes, a cycle length of $T = 200ms$, a data arrival rate λ at each node of 1 packet per second, and a queue capacity Q at each node of 10. For each set of simulations, we vary one of these parameters and investigate the throughput, delay and average energy consumption per second for X-MAC. The 95% confidence interval of the simulation results are also shown in the following figures. Fig. 7 shows the performance results under varying cycle length T from 50ms to 300ms. Fig. 8 shows the performance results varying the number of nodes N in the network from 5 to 40. Fig. 9 shows the performance results under varying data arrival rate λ from 0.5 packets per second to 2.5 packets per second. From these figures, we can see that our analytical results using the

Markov model match the simulation results.

In Fig. 7a, as the cycle length increases, the throughput of X-MAC first remains the same as the amount of input data, and then decreases as X-MAC can no longer deliver all the incoming packets. Since X-MAC delivers at most one packet in a cycle, a longer cycle length leads to lower throughput. In Fig. 8a, before X-MAC saturates, the throughput of X-MAC increases as the number of nodes in the network increases. However, as the number of nodes increases further, the throughput of X-MAC saturates and then shrinks slightly due to the increasing collisions. In Fig. 9a, as the data arrival rate increases, the throughput of X-MAC first increases linearly, and then remains the same when the queue at each node overflows. Since every node has a packet to send whenever it wakes up, the contention in the network remains the same. Therefore, the throughput of X-MAC does not change after saturation.

In Fig. 7b, when the cycle length is small, X-MAC can deliver all the incoming packets as soon as they arrive in the network, hence the delay of X-MAC is nearly zero. However, as the cycle length increases, X-MAC saturates and the queue at each node overflows. Both the contention delay and queuing delay, and hence the total delay of X-MAC, increases in proportion to the increase of the cycle length. In Fig. 8b, the delay of X-MAC remains very small before X-MAC saturates, and then increases as the number of nodes increases. This is because the more nodes in the network, the lower the probability for each node to transmit a DATA packet in a cycle, and the higher probability that the node has a longer queue. Hence, the contending delay and queuing delay of X-MAC increases. In Fig. 9b, the delay of X-MAC increases fast when X-MAC transits from the unsaturated region to the saturated region. This is because the queue at each node increases dramatically as X-MAC saturates. When the data arrival rates further increase, the increase of the delay slows down, as the queue length approaches the queue capacity.

In Fig. 7c, the energy consumption of X-MAC decreases as the cycle length increases. Although the energy consumed in data transmission increases as the cycle length increases, all the nodes that are not involved in the DATA transmission go to sleep for a longer time as the cycle length increases (the active period in each cycle is fixed). Since the energy savings in longer sleeping is more than the energy consumption in longer data transmissions, the energy consumption of X-MAC decreases as the cycle length increases. In Fig. 8c, X-MAC has decreasing energy consumption as the number of nodes in the network increases. Since X-MAC delivers at most one DATA packet in a cycle, more nodes in the network implies that

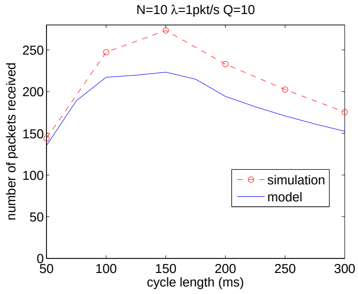


Fig. 10. X-MAC performance optimization under different cycle lengths.

a higher percentage of nodes fails to access the media and goes to sleep without consuming energy. Hence, the energy consumption of X-MAC decreases as the number of nodes increases. In Fig. 9c, the energy consumption of X-MAC increases when X-MAC can deliver all the incoming packets. Therefore, the more packets delivered, the higher energy consumption of X-MAC. However, when X-MAC saturates and the queue at each node overflows, the higher the data arrival rate is, the more packets are dropped by the queue. Since the contention in the network and the number of packets that are finally transmitted by X-MAC remains the same, the energy consumption of X-MAC is limited when X-MAC saturates.

E. X-MAC Optimizations

Our proposed Markov model and performance analysis can be used to optimize some protocol parameters to achieve desirable performance. In this section, we show an example of optimizing the cycle length of X-MAC in order to receive the most DATA packets in the lifetime of a network. The number of packets received at each node can be obtained by (43). Fig. 10 shows the number of packets a node can receive during its lifetime under different cycle lengths. Our model gives accurate prediction to the simulation results. For a fully-connected network with 10 nodes, if the data arrival rate is 1 packet per second at each node and the queue capacity is 10, the optimal cycle length is 150 in order to receive the most DATA packets at each node.

VII. Discussions

A. Model Extensions

The proposed Markov model and performance analysis assumes no retransmission, ideal channels and a maximum of one DATA packet transmission and one DATA packet reception per node per cycle. However, our model can be extended to support retransmission, non-ideal channels and multiple packet transmissions/receptions per node per cycle.

Specifically, to support R retransmissions, $R+1$ layers of our proposed Markov model are cascaded one after another, as shown in [8]. The $R+1$ empty-queue states are merged to one. Transition probabilities from one retransmission stage to the next higher level retransmission stage can be described as a function of the probability p_f of a DATA transmission failure and data arrival information $(A_i, A_{\geq i}, \lambda)$. In this case, π_0 can be obtained as a function of the probability p_s for each node to successfully transmit a DATA packet and the probability p_f of a DATA transmission failure. In the performance analysis, (p_s, p_f) is obtained as a function of π_0 according to the media access rules. Solving these functions between π_0 and (p_s, p_f), the throughput, delay and energy consumption of the investigated protocol can be determined following the calculations in Section III.

In terms of supporting non-ideal channels, our proposed Markov model works for deterministic channels, which can be modeled as having a constant probability of packet loss p_l. In the case of no retransmissions, no modification is needed to our proposed Markov model, however, the analytical throughput needs to be updated accordingly, by multiplying $(1-p_l)$ to equation (10). For random channels, the similar average p_l can be determined. However, randomness may incur large estimation errors as the variance of the channel increases. Hidden terminals can also be handled by our Markov queueing model. However, the function of $p = g(\pi_0)$ should consider the existence of hidden terminals, as shown in [25].

Our Markov model can also be extended to handle more than one packet transmission/reception per node per cycle. Given the maximum number M of packets that can be transmitted per cycle, transition probabilities from one state to another need to be updated as (1) the queue length could be reduced by up to M packets in a cycle, and (2) the queue length could be increased to up to Q packets in a cycle. Also, since more than one DATA packet can be transmitted in a cycle, the calculation of $p = g(\pi_0)$ changes accordingly (e.g., see the discussion of T-MAC in the next subsection). Solving $\pi_0 = f(p)$ obtained from our Markov model, and $p = g(\pi_0)$ obtained from the media access rules, p and π_0 can be determined, and the performance of the investigated protocol can be obtained.

B. Applying the Proposed Methodology to Other MAC Protocols

Our proposed Markov model and performance analysis methodology can be applied to not only S-MAC and X-MAC, but also other variations of duty-cycled MAC protocols. Here we briefly show a few examples of using our method to model different duty-cycled MAC protocols.

Among synchronized MAC protocols, T-MAC [6] greatly improves the throughput and delay of S-MAC by allowing more than one communication pair to send a DATA packet one after another within a cycle. This can be done by allowing every node to contend for the media after one pair of nodes in the network finishes a DATA transmission. To support T-MAC, our proposed Markov model remains the same. If a maximum of M packets can be transmitted within a cycle, the probability p for each node to transmit a DATA packet is the summation of the probability of being the 1st contention winner in a cycle and the probability of being the 2nd contention winner in a cycle, until the probability of being the Mth contention winner in a cycle. Those probabilities can be obtained similarly to obtaining p for S-MAC. Once $p = g(\pi_0)$ is obtained, the performance of T-MAC can be obtained by following the throughput, delay and energy consumption analysis in Section IV.

Based on T-MAC, the authors in [26] proposed ADV-MAC, which improves the energy consumption while keeping the high throughput and low delay of T-MAC. ADV-MAC introduces an advertisement period between the synchronization period and the data period in each cycle. In the advertisement period, every node with a packet to send broadcasts an ADV packet with its ID and its destination's ID. Hence, nodes who will not be involved in the data transmission in the data period can go to sleep directly. When the data period starts, a node with a packet to send will contend for the media by sending an RTS packet. The winner will start a DATA transmission, and the other nodes will sleep until the winner finishes its DATA transmission, and resume contending for the media. For ADV-MAC, the calculation of p as a function of π_0 is the same as T-MAC in a fully-connected network. However, the energy consumption of ADV-MAC is less then T-MAC, as nodes go to sleep when they have no packet to send/receive, or they are waiting for the contention winners to finish DATA transmissions.

For asynchronous duty-cycled MAC protocols, our proposed Markov model and performance analysis methodology can be applied to various protocols, such as CMAC [27], MX-MAC [28], and SpeckMAC [7]. CMAC is similar to X-MAC in terms of media access rules. It also has certain data routing function. Hence, for a fully-connected network, CMAC and X-MAC share the same Markov model and performance analysis.

Similar to X-MAC, in MX-MAC, instead of using preamble packets, a node sends DATA packets periodically until the receiver wakes up, receives the DATA, and then sends back an

ACK. Hence, the throughput and delay of MX-MAC can be obtained in the the same way as X-MAC using our proposed Markov model and performance analysis in Section VI. However, the energy consumption of MX-MAC can be obtained from the energy consumption of X-MAC by (1) making the length of a preamble packet the same as the length of a DATA packet, and (2) removing the energy consumed by the DATA packet transmission and reception in X-MAC, as preamble packets are DATA packets in MX-MAC.

SpeckMAC is similar to MX-MAC, except that an ACK is not required after the receiver successfully receives a DATA packets. Again, SpeckMAC and X-MAC use the same Markov model, and their throughput and delay can be obtained in the same way. The energy consumption of SpeckMAC can be further modified based on the energy consumption of MX-MAC by removing the energy consumed by listening for ACKs and sending an ACK.

Our proposed Markov model and performance analysis methodology may also be used in multi-channel duty-cycled MAC protocols. For example, [29] proposes a middleware between the MAC layer and the PHY layer. The middleware can be integrated with existing single-channel MACs, in order to dynamically share the spectrum for parallel communication. A receiver node wakes up once in a cycle, listens to the broadcast channel (for control messages), to the unicast channel (for DATA transmission) and to the candidate unicast channel (for channel hopping), alternatively. In this case, the cycle length for DATA transmission (3 times the real cycle length of the node) should be used in our Markov model and performance analysis. Meanwhile, the throughput obtained from our performance analysis should be multiplied by the frequency reuse gain, created by the multi-channel parallel communications.

Generally, our proposed performance analysis methodology includes 4 steps, (1) using the proposed Markov queuing model to obtain $\pi_0 = f(p)$, (2) obtaining $p = g(\pi_0)$ according to the media access rules of the investigated MAC protocol, (3) solving $\pi_0 = f(p)$ and $p = g(\pi_0)$ to obtain p and π_0 , and (4) using p and π_0 to calculate throughput, delay and energy consumption of the investigated protocol.

VIII. Conclusions

In this paper, we propose a Markov queueing model for duty-cycled nodes in wireless sensor networks, and then using the proposed model we analyze the throughput, delay, and energy consumption of duty-cycled MAC protocols with applications to S-MAC and X-MAC. Our

Markov model together with the performance analysis can (1) well estimate the throughput, delay and energy consumption for both synchronized and asynchronous duty-cycled MAC protocols, hence saving time by not requiring massive simulations to evaluate protocol performance, and (2) provide flexibility to achieve complex performance requirements by optimizing the protocol parameters. Another contribution of this work is that our proposed Markov queing model and performance analysis is the first to generalize synchronized and asynchronous duty-cycled MAC protocols for wireless sensor networks. Hence, it can be used to evaluate the performance of not only S-MAC and X-MAC as we illustrated in the paper, but also other existing or new synchronized/asynchronous duty-cycled MAC protocols, as shown in the Section VII. We show two examples (S-MAC and X-MAC) of how to apply our proposed Markov model to analyze the performance and optimize the protocol parameters. Comprehensive simulation results validate our model under various network conditions and application requirements.

REFERENCES

[1] F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovi, "Optimal transmission radius for energy efficient broadcasting protocols in as hoc and sensor networks", IEEE Trans. Parallel and Distributed System, vol. 17, no. 6, Jun. 2006, pp. 536-547.

[2] H. Luo, J. Luo, Y. Liu, and S. Das, "Adaptive data fusion for energy efficient routing in wireless sensor networks", IEEE Trans. Computers, vol. 55, no. 10, Oct. 2006, pp. 1286-1298.

[3] Y. Wei, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks", in Proc. INFOCOM'02, pp. 1567-1576.

[4] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks", in Proc. SenSys'04, pp. 95-107.

[5] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks", in Proc. SenSys'06, pp. 307-320.

[6] T. Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks", in Proc. 1st Int. Conf. on Embedded Networked Sensor Systems, 2003, pp. 171-180.

[7] K. Wong and D. Arvind, "SpeckMAC: Low-power decentralised MAC protocol low data rate transmissions in Specknets", in Proc. REAL-MAN06.

[8] O. Yang and W. Heinzelman, "Modeling and Throughput Analysis for SMAC with a Finite Queue Capacity", in Proc. ISSNIP'09, pp. 409-414.

[9] O. Yang and W. Heinzelman, "Modeling and Throughput Analysis for XMAC with a Finite Queue Capacity", to appear in Proc. GlobeCom'10.

[10] J. Zhang, F. Nat-Abdesselam, and B. Bensaou, "Performance Analysis of an Energy Efficient MAC Protocol for Sensor Networks", in Proc. Int. Symposium on Parallel Architectures, Algorithms, and Networks, 2008, pp. 254-259.

[11] S. Hong and H. Kim, “A Multi-hop Reservation Method for End-to-End Latency Performance Improvement in Asynchronous MAC-based Wireless Sensor Networks”, IEEE Trans. Consumer Electronics, vol. 55, no. 3, Aug. 2009, pp. 1214-1220.

[12] K. Kumar and P. Kumar, “Tmote implementation of BMAC and SMAC protocols”. Available: http://www.cse.iitk.ac.in/users/vkirankr/wireless_report.pdf.

[13] V. Tippanagoudar, I. Mahgoub, and A. Badi, “Implementation of the Sensor-MAC protocol for the JiST/SWANS simulator”, in Proc. IEEE/ACS Int. Conf. on Computer Systems and Applications, 2007, pp. 225-232.

[14] G. Bianchi, “IEEE 802.11 saturation throughput analysis”, IEEE Communication Letters, vol. 1, no. 12, Dec. 1998, pp. 318-320.

[15] J. He, Z. Tang, H. Chen, and Q. Zhang, “An accurate and scalable analytical model for IEEE 802.15.4 slotted CSMA/CA networks, in IEEE Trans. Wireless Communications, vol. 8, no. 1, Jan. 2009, pp. 440-448.

[16] Y. Zhang, C. He, and L. Jiang, “Energy and QoS Trade-off Analysis of S-MAC protocol in Wireless Sensor Networks”, in Proc. CCWMSN’07, pp. 76-79.

[17] Y. Wang, M. Vuran, and S. Goddard, “Cross-layer Analysis of the End-to-end Delay Distribution in Wireless Sensor Networks”, in Proc. 30th IEEE Real-Time Systems Symposium, 2009, pp. 138-147.

[18] C. Fischione, S. Ergen, P. Park, K. Johansson, A. Sangiovanni-Vincentelli, “Medium Access Control Analytical Modeling and Optimization in Unslotted IEEE 802.15.4 Wireless Sensor Networks”, in Proc. SECON’09, pp. 1-9.

[19] J. Rousselot, A. El-Hoiydi, and J. Decotignie, “Low Power Medium Access Control Protocols for Wireless Sensor Networks”, in Proc. 14th European Wireless Conference, 2008, pp.1-5.

[20] J. Luo, L. Jiang, and C. He, “Finite Queuing Model Analysis for Energy and QoS Tradeoff in Contention-Based Wireless Sensor Networks”, in Proc. ICC’07, pp. 3901-3906.

[21] J. Luo, L. Jiang, and C. He, “Performance Analysis of Synchronous Wakeup Patterns in Contention-based Sensor Networks Using a Finite Queuing Model”, in Proc. GlobeCom’07, pp. 1334-1338.

[22] Y. Fallah, F. Agharebparast, M. Minhas, H. Alnuweiri, and V. Leung, “Analytical modeling of contention-based bandwidth request mechanism in IEEE 802.16 wireless networks”, IEEE Trans. Vehicular Technology, vol. 57, no.5, Sep. 2008, pp. 3094-3106.

[23] Available: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf.

[24] C. Merlin, “Adaptability in Wireless Sensor Networks Through Cross- Layer Protocols and Architectures, Ph.D. Dissertation, Dept. ECE, University of Rochester, Rochester, NY, 2009.

[25] O. Yang and W. Heinzelman, “Modeling and Throughput Analysis for SMAC with Hiddern Terminals”, Technical Report, University of Rochester, 2010. Available: https://urresearch.rochester.edu/viewResearcherPage.action?researcherId=100.

[26] S. Ray, I. Demirkol, and W. Heinzelman, “ADV-MAC: Analysis and Optimization of Energy Efficiency through Data Advertisements for Wireless Sensor Networks”, Ad Hoc Networks, 2010, doi:10.1016/j.adhoc.2010.10.005.

[27] S. Liu, K. Fan, and P. Sinha, “CMAC: An energy efficient MAC layer protocol using convergent packet forwarding for wireless sensor networks”, in Proc. SECON’07, pp. 11-20.

[28] C. Merlin and W. Heinzelman, “Schedule Adaptation of Low-Power-Listening Protocols for Wireless Sensor Networks”, IEEE Trans. Mobile Computing, vol. 9, No. 5, May 2010.

[29] G. Zhou, L. Lu, S. Krishnamurthy, M. Keally, and Z Ren, “SAS: Self-Adaptive Spectrum Management for Wireless Sensor Networks”, in Proc. ICCCN’09, pp. 1-6.

Ou Yang Ou Yang received a B.S. degree in 2003 in Telecommunications Engineering, from Beijing University of Posts and Telecommunications, Beijing, China, and a M.S. degree in 2006 in Electronics Engineering, from Tsinghua University, Beijing, China. She is currently working toward a Ph.D. degree at the Department of Electrical and Computer Engineering, University of Rochester, USA. Her current research interests include wireless communications and networking, and low-power solutions for wireless sensor networks.

Wendi Heinzelman Wendi B. Heinzelman is an associate professor in the Department of Electrical and Computer Engineering at the University of Rochester, and she holds a secondary appointment as an associate professor in the Department of Computer Science. Dr. Heinzelman also currently serves as Dean of Graduate Studies for Arts, Sciences and Engineering at the University of Rochester. Dr. Heinzelman received a B.S. degree in Electrical Engineering from Cornell University in 1995 and M.S. and Ph.D. degrees in Electrical Engineering and Computer Science from MIT in 1997 and 2000, respectively. Her current research interests lie in the areas of wireless communications and networking, mobile computing, and multimedia communication. Dr. Heinzelman received the NSF CAREER award in 2005 for her research on cross-layer architectures for wireless sensor networks, and she received the ONR Young Investigator Award in 2005 for her work on balancing resource utilization in wireless sensor networks. She is an Associate Editor for the IEEE Transactions on Mobile Computing, an Associate Editor for the ACM Transactions on Sensor Networks and an Associate Editor for Elsevier Ad Hoc Networks Journal. Dr. Heinzelman is a senior member of the IEEE and the ACM, and she is co-founder of the N^2 Women (Networking Networking Women) group.