Impact of Sleep in Wireless Sensor MAC Protocol

Subah Ramakrishnan, Hong Huang, Manikanden Balakrishnan
Klipsch School of Electrical Engineering
New Mexico State University
Las Cruces, USA
{ramakris, hhuang, mabalakr}@nmsu.edu

John Mullen
Industrial Engineering
New Mexico State University
Las Cruces, USA
{jomullen}@nmsu.edu

***Abstract*—A sensor net consists of tiny devices with severe energy constraints. Employing energy efficient, sensor-specific MAC protocols is of necessity for sensor nets, and such protocols have been proposed in the literature. Sensor MAC protocols introduces sleep in sensor nodes to conserve energy and the impact of sleep on protocol performance needs a detailed investigation. In this paper, we quantify the effect of sleep on protocol performance through queueing analysis and simulation. Results demonstrate quantitatively the tradeoff between energy consumption and average latency in sensor MAC protocols.**

Keywords-Media Access Control; Sensor Networks

I. INTRODUCTION

Wireless sensor nets have wide-ranging applications such as environment protection, homeland security and digital warfare. A wireless sensor net consists of many tiny devices, powered by small-sized batteries, and operates unattended for prolonged duration. Energy conservation is crucial since the life time of a sensor net is determined by its power consumption rate. Conventional Media Access Control (MAC) protocols, e.g., IEEE 802.11 Distributed Coordinated Function (DCF), are not suitable for sensor networks, for they are not optimized to conserve energy.

Several energy conserving MAC protocols have been proposed. The IEEE 802.15.4 [1] (under development) is a sensor MAC protocol that uses periodic sleep in nodes to conserve energy and requires synchronization to decide on suitable schedules. TMAC is another example that introduces sleep in nodes to reduce energy consumption [2]. This protocol dynamically maintains an active period where the nodes transmit data in bursts and switches to sleep state between bursts on detecting no activity until time-out. TMAC also requires synchronization to reduce latency caused by sleep. An energy-efficient MAC protocol (SMAC) is proposed in [3]. SMAC modifies DCF by putting nodes to sleep at certain times to conserve energy, but this also incurs delays caused by sleep.

The main objective of this paper is to investigate the effects of sleep delay on protocol performance. In our study, we base our sensor MAC modeling mostly on SMAC [3], because SMAC is well designed, is representative of issues in sensor MAC protocols, and has a *ns* [4] simulation module available. In the paper, an approximate queueing model for SMAC is developed and analyzed for Poisson traffic; throughput and latency equations for non-saturated conditions are derived to statistically analyze sleep delay. The analytical model is validated by simulations in *ns* [4]. The paper is organized as follows. Section II outlines the SMAC protocol. Section III provides a detailed description of the queueing model for SMAC. Section IV provides the performance evaluation of the protocol. Section V concludes and points out future work.

II. SMAC PROTOCOL DESCRIPTION

A detailed description of SMAC protocol can be found in [3], here we summarize briefly. Three major energy wastage events occurring at a conventional MAC layer have been identified in [3]: (a) Collision results in energy wastage due to retransmissions of collided packets. (b) Overhearing occurs when a node listens to transmissions not intended for it, needlessly wasting power. (c) Idle listening occurs when nodes listen in the hope of receiving any possible data, also wasting energy. Since the power consumed by nodes in idle, receive and transmit states are on the same order of magnitude, the power wastage caused by overhearing and idle listening is no less serious than that of collisions. The main idea of SMAC is to put nodes to sleep from time to time to reduce energy wastage caused by the above events. A node goes to sleep periodically if it is not engaged in transmission or reception, to reduce idle listening. It also goes to sleep if its neighbors are involved in communication, of which it is not a party, to reduce collision and overhearing.

A *cycle* in SMAC consists of a listen and a sleep state. A node normally follows predetermined schedules to wake up or go to sleep with the following exceptions: (i) A node goes to sleep if any of its neighbors are communicating, and the node is not a party. (ii) A node wakes up at the end of its neighbor's transmission if it needs to relay the packet. This is done by overhearing neighbor's RTS and CTS exchanges before the node goes to sleep and serves the purpose of reducing latency caused by sleeping. This behavior is called adaptive listening [3].

Schedules are periodically exchanged by broadcasting SYNC packets among neighboring nodes to induce synchronized listen behavior as much as possible and thus to reduce latency caused by sleeping. Synchronized neighbors form a *virtual cluster*, but synchronization can only be achieved to a certain extent in an ad hoc environment. Lack of complete synchronization introduces sleep delay which results in increased packet latency [3].

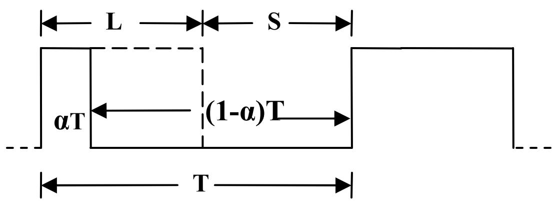
III. QUEUEING MODEL FOR SMAC

We consider a system composed of *N* interfering nodes. Traffic arrival is assumed to be Poisson with a rate being λ packets per node per slot time, which approximates a class of application where events to be sensed arrive randomly. Therefore the aggregate arrival rate to the channel is $N\lambda$. Channel service rate is denoted by μ and is defined as the number of packets serviced per slot time by the shared channel. The service time is calculated as the sum of the following delay components, which include: (a) sleep delay due to lost transmission opportunity caused by sleep, (b) contention delay or the time spent to win contention, and (c) transmission delay, which includes time for retransmission. The service time follows a general distribution and the shared channel is represented by an M/G/1 queueing model. The following section presents a detailed discussion on sleep delay encountered by a packet.

A. *Sleep Latency:*

Sleep delay can be incurred in two distinct situations depending on whether the next packet to be served is (a) a new arrival or (b) from the queue.

In case (a), an incoming packet sees an empty queue. A packet is serviced in the current cycle only when it arrives within the current contention window (αT), otherwise it has to wait for the next cycle, refer to Figure 1.



T – *Cycle Time* **S** – *Sleep Time* **L** – *Listen Time*

α – *Fraction of time spent in contention*

Fig.1 The sleep/listen cycle.

Assuming random packet arrival time, the event that an arriving packet sees an empty queue yet still suffers a sleep delay is caused by the unfortunate combining of two independent events: empty queue and missing contention period. The probability of such event is thus given by:

$$p_1 = (1-\rho)(1-\alpha) \tag{1}$$

where ρ is equal to $N\lambda/\mu$ (queuing intensity per node), which is the probability that a node's queue is non-empty, and *T* is the total cycle time.

Assuming the packet arrives at any time instant equally likely after the contention period, the delay caused by sleep can be calculated as:

$$S_1 = \frac{(1-\alpha)T}{2} \tag{2}$$

In case (b), the sleep delay can be avoided if adaptive listening causes the next hop node, by overhearing neighbor's RTS/CTS exchanges, to wake up in time to relay the packet queued and scheduled to transmit from the previous hop node. Unfortunately, adaptive listening works only in alternate hops [3], since sleeping will cause a node to miss its neighbor's RTS/CTS exchanges. Considering the effect of adaptive listening, the probability that an incoming packet sees a non-empty queue and encounters a sleep delay is given as:

$$p_2 = \beta\rho \tag{3}$$

where β represents the fraction of hops traversed by a packet that does not encounter sleep delay, and is given by:

$$\beta = \frac{\lceil h/2 \rceil}{h} \tag{4}$$

where h is the number of hops traversed from the source to destination.

The delay encountered in this case can be calculated as:

$$S_2 = \frac{(1-\alpha)T}{2} \tag{5}$$

The overall sleep latency is given by,

$$S = p_1 S_1 + p_2 S_2 \tag{6}$$

B. *Total Latency:*

In addition to sleep delay, a packet also suffers from contention delay and transmission delay, which are computed as follows.

Contention delay is the time a node spends to win contention, which is also called channel access delay. The number of times a node will attempt to contend for the channel before success in a given backoff stage, is a geometrical random variable with a probability 1/C and is defined in [5]. Thus the expectation of the total time required to win contention is given by,

$$C = \frac{1 + W + pW\sum_{i=0}^{m-1}(2p)^i}{2} \tag{7}$$

where, *W* denotes minimum contention window size and *m* is the maximum number of backoff stages. The probability of packet collision, *p,* is defined as the probability that two or more nodes transmit in the same slot time and is derived in [5] as:

$$p = 1 - \left(1 - \frac{1}{C}\right)^{(N-1)} \tag{8}$$

where N is the number of interfering nodes. Equations (6) and (7) can be solved numerically to obtain the values of p and C.

Transmission delay (T) is just the time for the radio to transmit a packet, which is a function of channel data rate.

The total service time is given as:

$$\frac{1}{\mu} = S + C + T \tag{9}$$

and the overall latency encountered by a packet is given by the sum of service time and the queuing delay obtained for an M/G/1 system by applying the Pollaczek-Khinchin formula [6]. The average latency is written as:

$$L = \frac{1}{\mu} + \frac{N\lambda\left(\sigma_s^2 + 1/\mu^2\right)}{2\,(1-\rho)} \tag{10}$$

Where, σ_s^2 is the variance of the service time distribution and ρ is equal to $N\lambda/\mu$.

IV. PERFORMANCE EVALUATION

Numerical results are obtained using the formulation described in the previous section and Table I lists the important parameters used in the analysis.

TABLE I
PARAMETERS USED IN ANALYSIS

Channel Bandwidth	20Kbps
N	5
Data Packet size	50Bytes

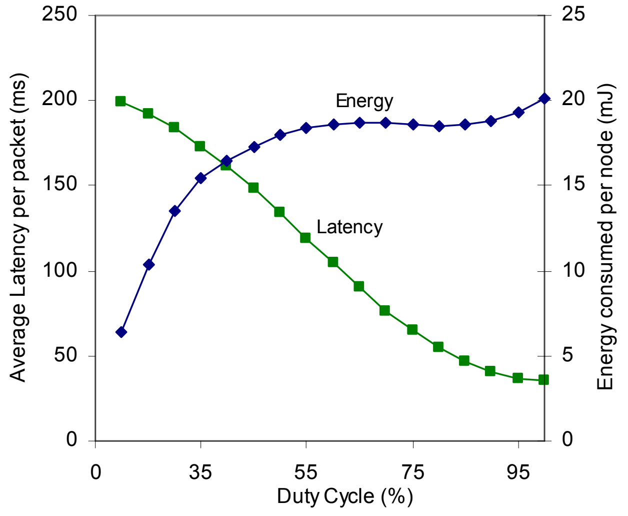


Fig.2 Latency and energy consumption results of SMAC obtained from queueing modeling

Figure 2 shows the performance of SMAC for varying duty cycle values. Duty cycle is defined as the fraction of total cycle time that a node listens, i.e., L/T in Figure 1.From Figure 2, as expected, average latency per packet is high at low duty cycles, because nodes sleep for longer duration of time and introduce large sleep delay. However energy consumed by a node increases with duty cycle since the node idlly listens for extended period of time.

A. *Simulation Environment:*

To validate our results, we simulated the performance of SMAC in *ns* [4]. A simple five-node network topology was used. Four nodes generate exponentially distributed traffic to a single sink node. Simulation parameters are listed in table 2, which are consistent with those from [3].

TABLE II
SIMULATION PARAMETERS

Channel bandwidth	20 kbps
Average packet size	50 Bytes
RTS, CTS, ACK size	30 Bytes
Reception power	13mW
Transmission power	24.75mW
Idle power	13mW
Sleep power	15µW

For the same network scenario, average energy consumption and latency obtained from analytic modeling were compared with simulation results. Fig 3 shows the results for average latency per packet at varying duty cycles. At 95% confidence intervals, it shows the simulation and analytical results are in reasonably good agreement. Other simulation data points show similar pattern, but are not included for the clarity of the figure. The figure shows that at low duty cycle, i.e., a node sleeps for a longer duration, the difference in packet latency for different arrival rates is large. This is because, at high arrival rates the demand for the channel is

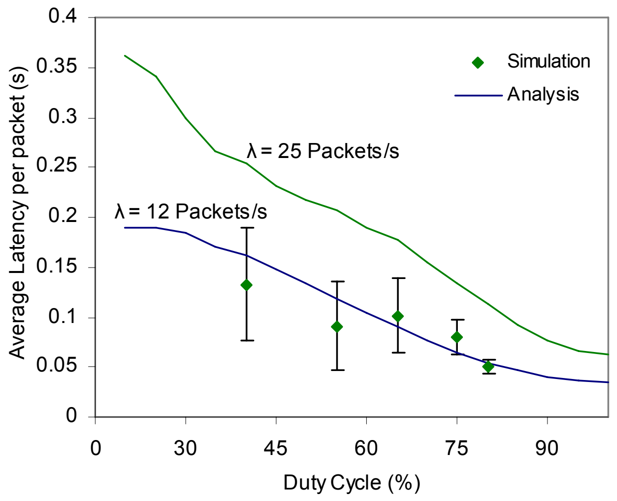


Fig.3 Latency results for SMAC obtained from queuing analysis and simulation

much higher than that at low arrival rates, therefore the performance is degraded much more at high arrival rates. As the duty cycle increases, the difference in packet latency for low and high arrival rates tends to disappear. The figure reaffirms the intuition that low duty cycle operation is appropriate for low arrival rates but can cause excessive latency at high arrival rates.

Fig 4 shows the results for average energy consumption obtained using analysis and simulation, respectively. Again the simulation results are at 95% confidence interval. The figure shows that the differences in the average energy consumption for different arrival rates increases as duty cycle increases. This is because, at low duty cycle, sleep behavior dominates energy consumption. As the duty cycle increases, packet transmission tends to dominate energy consumption. Therefore, low duty cycle operation is an effective way to limit energy consumption regardless of the traffic load.

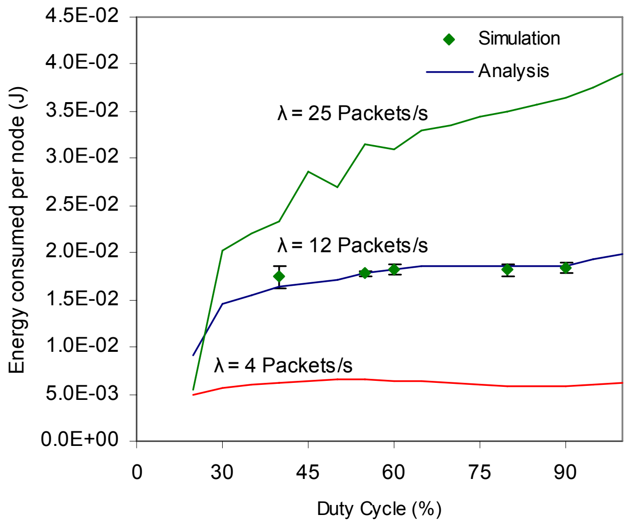


Fig.4 Energy consumption results for SMAC obtained from queuing analysis and simulation

V. CONCLUSIONS AND FUTURE WORK

In this paper, we quantified the performance impact of sleep in a sensor MAC protocol by queuing analysis and simulation. Our results demonstrate the tradeoff between latency and energy consumption under varying duty cycles and for different packet arrival rates. As future work, we plan to study the performance impacts of sleep on the nodes that play different roles in the network such as ordinary, gateway, cluster head nodes, etc.

REFERENCES

[1] Gutierrez, Jose et al., "IEEE 802.15.4: A Developing Standard for Low-power Low-cost Wireless Personal Area Networks," *IEEE Network Magazine, vol. 15, no.5, October 2000, pp.12-19.*

[2] Tijs Van Dam, Koen Langendoen, "An Adaptive Energy Efficient MAC protocol for Wireless Sensor Networks," *ACM SenSys '03, November 2003.*

[3] Wei Ye, John Heidemann, Deborah Estrin, "Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks", *Technical Report ISI-TR-567, USC/Information Sciences Institute, January 2003.*

[4] UCB/LBNL/VINT Network Simulator - ns (Version 2). *http://www-mash.cs.berkeley.edu/ns/.*

[5] Giuseppe Bianchi, "Performance Analysis of IEEE 802.11 Distributed Coordinated Function", *IEEE Journal on Selected Areas in Communications 18 (3), pp. 535-547, March 2000.*

[6] R. K. Jain, *The Art of Computer Systems Performance Analysis*: *Techniques for Experimental Design, Measurement, Simulation, and Modeling*, John Wiley and Sons, 1992.