Medium Access Control With A Dynamic Duty Cycle For Sensor Networks

Peng Lin, Chunming Qiao and Xin Wang
Department of Computer Science and Engineering
State University of New York at Buffalo
Buffalo, NY 14260
Email: {penglin,qiao,xwang8}@cse.buffalo.edu

Abstract— **Energy conservation is a primary concern in sensor networks. Several MAC protocols have been proposed to address this concern. However, the tradeoff between power consumption and latency has not been thoroughly studied. In this paper, we propose a sensor medium access control protocol with dynamic duty cycle, DSMAC, which achieves a good tradeoff between the two performance metrics without incurring much overhead. Moreover, DSMAC is able to adjust its duty cycle with varying traffic conditions without assuming any prior knowledge of application requirements. Both analytical and simulation results have been presented in this paper.**

I. Introduction

Recent advances in the wireless sensor network technology [1] have made various sensor network applications possible. A sensor node is a small device which is able to carry out distributed data gathering, processing and communications. A sensor network has a unique feature which distinguishes it from other wireless networks: every sensor in general has only limited power resources, processing capability and amount of storage.

A. Related Work

Due to the specific energy constrained environment, MAC design for sensor networks generally has to take energy consumption as one of its primary concerns. However, existing wireless medium access control protocols, such as bluetooth [2] or 802.11 [6] MAC protocols, cannot be applied directly to sensor networks, since none of them take energy conservation as a major objective.

There have been several MAC protocols specially designed for sensor networks. In [7], an adaptive transmission rate control (ARC) protocol has been proposed, which addresses the MAC fairness issue by balancing the originating traffic and route-through traffic. The Self-organizing MAC for Sensor networks(SMACS) protocol proposed in [3] adopts a random wakeup schedule. The most relevant work to the paper is the Sensor-MAC (SMAC) protocol proposed in [8], [9]. The coordinated sleeping mechanism used in SMAC is similar to that used in the power saving mode of 802.11. As analyzed in [9], SMAC explicitly trades latency for energy savings.

SMAC integrates the following features into a general contention-based MAC protocol to accomplish the goal of saving energy:

- Periodic Listen and Sleep: For general sensor network applications, sensing activities may be bursty. If sensors stay in an inactive state for a long duration listening for potential communications, energy will be unnecessarily wasted under such circumstances. SMAC divides time into frames and each frame into two periods: *listen* and *sleep*. Figure 1 illustrates the frame structure. The duty cycle is defined as the ratio of the listen interval to the frame length. The listen interval is further divided into smaller intervals dedicated to SYNC, RTS and CTS packets. The listen interval is normally fixed according to physical-layer and MAC-layer parameters.
- Collision Avoidance: Carrier sensing and RTS/CTS handshaking schemes were adopted in SMAC to solve the hidden terminal problem for data packets transmissions.
- Maintaining Synchronization: Each sensor maintains a schedule table, which is composed of neighbor schedules: time difference between itself and its neighboring nodes. SYNC packets are periodically broadcasted to maintain synchronization among the neighboring nodes. Upon the reception of an SYNC packet from its neighbor, the sensor will update the relevant entry in the schedule table such that the two nodes are synchronized. SYNC packets do not involve handshaking signalling. Synchronization has been a key requirement for all sensor networks where coordinated sleep schemes are adopted. There have been implementations of sensor networks, such as Mica mote platform [4] etc., which can achieve microsecond accuracy which is sufficient for most of the sensor applications.
- Schedule Determination: A schedule is a value indicating the time left from now before switching to the sleep state, based on a local clock. There is an initialization phase for sensors to either decide its own schedule, or simply follow the clock of one of its neighbors. The latter is preferred as the cost of maintaining clocks will be reduced.

B. Motivation of Our Work

"To sleep or not to sleep" is the question that motivated our work. More specifically, even though the static low duty cycle assignment in SMAC [8], [9] has been efficient in terms of energy conservation, the latency issue is neglected. In sensor

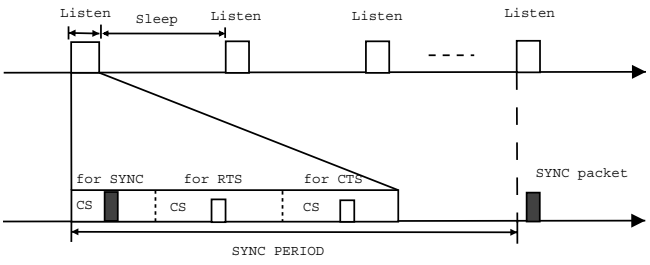

Fig. 1. SMAC Frame Structure

networks, latency has been a key factor affecting the applicability of sensor networks to some delay-sensitive applications, such as those used in health and military areas. Although it has been stated in [9] that duty cycle can be manually changed with application requirement knowledge, we believe that a generally designed dynamic SMAC protocol, which does not assume any prior application requirement knowledge but can tune its duty cycle automatically is needed. Based on such observations, we develop the Dynamic Sensor MAC protocol (DSMAC) with a dynamic duty cycle in this paper. DSMAC is able to dynamically change the sleeping interval with fixed listen interval length and therefore the duty cycle of sensors is adjusted to adapt to the current traffic condition. Therefore, DSMAC alleviates the high latency problem presented in SMAC when the traffic load is high, while still keeping the energy efficiency when the traffic load is low. Moreover, DSMAC only introduces insignificant overhead than SMAC.

The rest of this paper is organized as follows. Section II shows the DSMAC design, followed by a brief analysis on the latency performance of DSMAC in Section III,. Section IV demonstrates the energy consumption and latency tradeoffs achieved by DSMAC through simulations. Finally, we conclude the paper in Section V.

II. DSMAC Design

In this section, a brief outline of the DSMAC protocol is given first, followed by a detailed description of each component of DSMAC.

A. Synchronization

With DSMAC, the nodes form groups of peers, and each node uses the SYNC packets to deal with clock drift. More specifically, upon the deployment, a sensor first carries out a scanning phase, the sensor listens to the channel for a certain period and attempts to adopt an existing sleep-wakeup schedule enclosed in a SYNC packet if any, as is done similarly in SMAC. However, if no SYNC packet has been received at the end of the period, the sensor node assumes that it is the first active sensor in its perceivable neighborhood and chooses a schedule freewillingly. The node then notifies forthcoming nodes its schedule through periodically broadcasting SYNC packets. Each sensor maintains a synchronization table for its neighboring nodes. Once the sleep-wakeup schedule is decided, every node will start broadcasting a SYNC packet which contains the clock synchronization information. Whenever a sensor overhears a SYNC packet, it will update its synchronization table and adjust its timer according to the one of the SYNC packet originator's.

B. Algorithm Description

In this section, we describe the DSMAC algorithm in detail. Unless otherwise stated, the packet formats used in this paper refer to those defined in SMAC [8]. As discussed above, the constant duty cycle design in SMAC trades off latency for saving the energy consumption. Thus while it is efficient for energy utilization, SMAC is not suitable for delay-sensitive sensor applications which require a prompt response from sensor devices.

The proposed DSMAC protocol attempts to adjust its sleep-wakeup cycle time dynamically according to the current energy utilization efficiency and average latency experienced by the sensor. We define one-hop latency as the difference between a packet gets into the queue and it is successfully sent out. The value is recorded in the packet header by the sending sensor node and retrieved by the receiving node. The average latency of the receiving sensor is the average value of all one-hop latency values collected in the current SYNC period. This average latency value serves as an approximate estimation of the current traffic condition and an indicative parameter for the receiving node. Since the delay value is measured at each sending node, there is no synchronization issue involved. The DSMAC protocol achieves the above mentioned goal through a two-phase tuning procedure. In a DSMAC-enabled network, every sensor has assumed all functionalities defined in the SMAC protocol. Aside from that, each receiver node also keeps track of its own energy consumption efficiency and average latency. The energy consumption level can be measured by monitoring the energy consumption amount per packet delivered.

One main reason that SMAC adopted a constant duty cycle is the difficulty in maintaining synchronization among the sensors in a network. With a constant duty cycle, multiple sensors are able to share one common sleep-wakeup cycle and hence fewer schedules need to be synchronized. DSMAC addresses the issue as follows: All sensors adopt a common basic service duty cycle at the beginning. For a receiver sensor node, if it notices that the latency becomes intolerable, it unilaterally makes a decision to double the original duty cycle by shortening the sleep time period length accordingly without changing the listen time period. Therefore, the node which increases its duty cycle is able to get more chances to receive packets instead of blocking other senders while sleeping. Moreover, the SYNC packet has also contained a new field indicating the SYNC initiator's duty cycle. The node broadcasts its current duty cycle via the SYNC packet in addition to synchronization information. Neighboring sensor nodes which have overheard the SYNC packet will check its own queue, and if it has packets to send and the duty cycle specified in the SYNC packet is higher than its existing schedule, then it will double its own duty cycle. Otherwise, if it is a new schedule, the receiver will simply update its schedule

table. More specifically, let T_E value represent a threshold which indicates the upper bound on the energy consumption level. Only when the current power consumption level is below the T_E value, doubling duty cycles is allowed. A more detailed description of the MAC has been given in Algorithms 1-3. In the algorithm descriptions, we assume that only the following duty cycles are used: 10%, 20% and 40%, which are empirically chosen based on the SMAC implementation in [8].

Algorithm 1 handleDATA

Upon the reception of a data packet, get the delay based on the timestamp information marked by the sender of the packet. Update its average packet delay, power consumption and packet counter for the current SYNC period.

Algorithm 2 sendSYNC

Upon the SYNC packet initiating time, construct a SYNC packet.
if ((My queue is empty || $avg\_delay < Dmin$) && duty cycle $> 10\%$) **then**
  Halve my current duty cycle and put my new duty cycle into the SYNC packet.
end if
if (0 $<$ current energy consumption level $< T_E$ && $avg\_delay > Dmax$ && my duty cycle $< 40\%$) **then**
  Double my current duty cycle and put my new duty cycle into the SYNC packet.
end if

Algorithm 3 handleSYNC

Upon the reception of an SYNC packet time, construct a SYNC packet.
if (find the schedule of the SYNC sender) **then**
  if (My queue is not empty && the duty cycle in the SYNC packet $>$ my duty cycle) **then**
    Set my duty cycle according to the duty cycle given in the SYNC packet
  end if
end if
if (the schedule of the SYNC sender is not found) **then**
  Add the schedule in the SYNC packet to my schedule list
end if
Update the schedule table

One of the main merits of the DSMAC's dynamic duty cycle adjustment scheme is its scalability, in the sense that the adjustment action does not affect other neighbors' schedules if they are idle, and hence the cost incurred due to the adjustments will not increase drastically with respect to nodal density. The reason lies in that the increased duty cycle is always a multiple of the basic duty cycle increasing exponentially. Therefore, the original listen period in the old schedule is still a listen period in the new schedule. The adjustment procedure will not cause any synchronization problem for the nodes that are not involved. This is important in synchronization-based sensor networks. Imagine the following scenario: suppose node B, a direct neighbor of node A, has adjusted its duty cycle with its other neighbors and A has been lightly loaded and not been involved in the duty cycle adjustment, there should not be any synchronization problem between node A and B. Node A can still use its own schedule to communicate with its neighbors, including B. The only difference is that node B now wakes up more often than before. However, node A does not need to be aware of it, as the original listen time of B is still a listen time, despite the new duty cycle. An example illustration is given in Figure 2.

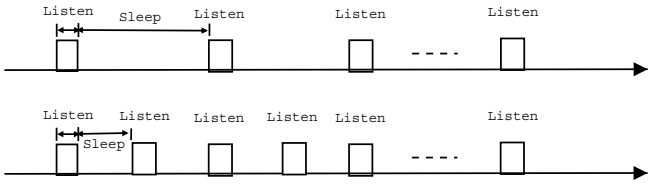

Fig. 2. Neighboring nodes which adopt different duty cycles can still communicate with old schedule

As shown in Algorithm 2, a node may need to reduce its duty cycle by halving under certain circumstances. There may be cases when a node satisfies both doubling and halving criteria. For example, if a receiver B decides to halve its duty cycle as in algorithm 2 while the sender A wants to send B packets and hopes B can wake up more often. B may not be able to accommodate A's request promptly. In this case, B will double its duty cycle after the reception of the next SYNC packet from A.

The protocol overhead introduced by DSMAC contains an extra "duty cycle" field in the existing SYNC packet and a "delay" field for data packets which can be negligible. Compared with the SMAC implementation [8], each DSMAC sensor also needs to maintain its own average delay and energy efficiency level. However, the processing and storage overhead introduced here can be also negligible and all these overheads are well compensated by the reduced queueing cost due to the decreased latency.

III. Latency Analysis

In this section, we analyze the latency performance of DSMAC protocol, and compare it with the SMAC protocol.

First, we describe the terminologies used in this paper.

- Carrier sense delay: latency incurred due to the carrier sensing procedure when a sensor contends for the channel. We denote the average value of the carrier sense delay by t_{cs}, which is determined by the contention window size. In DSMAC, the contention window size is fixed, and so is the t_{cs}.
- Backoff delay: latency incurred due to carrier busy or collision recovery procedure. When the sensor node is performing a carrier sensing operation and detects a busy channel, it carries out a backoff procedure.

- Transmission delay: latency related to channel bandwidth, packet length and the coding scheme. Manchester encoding is adopted as the coding scheme in this paper. We assume a fixed packet length and denote the transmission delay by t_{tx}.
- Sleep delay: latency incurred due to the periodic sleeping of each sensor. This latency component is uniquely presented in SMAC/DSMAC, as both of them enforce the periodic sleeping algorithm at each sensor node. It is denoted as t_s in the context.

1) Single-hop Case: In this subsection, we look at the latency for a packet transmission between two neighboring nodes that are one hop away from each other. Due to the channel capacity limitation, we ignore the propagation delay and the processing delay. Since queueing delay and backoff delay appear in any contention based CSMA MAC protocols, we do not take into account those components and focus on the latency components which are more specific to DSMAC: carrier sense delay, transmission delay and sleep delay.

In SMAC [8], a latency analysis has been given on a simple light traffic load case under the assumption that a packet always arrives during the listen time interval and therefore, it will be forwarded to its next hop destination without experiencing any queueing delay. This assumption is valid only for light traffic load scenarios. In this paper, we argue that DSMAC is able to dynamically adjust the communication peers' sleeping cycles regardless of the traffic load. Therefore, in our analytical model, no specific traffic load assumption has been made. It should be noted that SMAC proposed a scheme named adaptive listen which attempted to shorten the end to end multi-hop delay by forcing the next two-hop away neighbor, which is a potential next-hop neighbor for the current receiver node, to standby after the transmission between the current peers is done. This mechanism is able to reduce the latency accumulation effect for multi-hop transmissions. However, it does not alleviate the latency caused by one-hop transmissions. For one-hop packet transmissions, adaptive listening makes no impact since no two-hop neighbor is involved.

In the following analysis, we consider the single-hop latency defined as the time duration between the time when a sensor node has a head-of-line packet ready to send and the packet gets delivered to its one-hop neighbor. As discussed above, the queueing delay and backoff delay components are ignored here.

a) per-hop delay without sleeping: In SMAC, the per-hop delay for scenarios, where every sensor is working in full duty, can be calculated as follows:

$$D = t_{cs} + t_{tx} \tag{1}$$

This scenario assumes a full duty cycle, as the system resource has been exhausted at the maximum rate. It is a special case of SMAC with 100% duty cycle. In DSMAC, duty cycle is not fixed, and hence, the scenario does not apply.

b) per-hop delay with sleeping: As mentioned above, in our analytical model, we do not make any assumption on traffic load. For other parameters, we adopt the same notations as in SMAC, where T_f represents the length of a complete sleep-wakeup period. We can get the latency of a single-hop communication experienced in the network given as below:

$$D = t_{cs} + t_{tx} + t_s \tag{2}$$

The sleep delay t_s is a random variable between (0, T_f). For SMAC, it has been given in [9]:

Suppose t_s is uniformly distributed in (0, T_f), we have the average latency for one-hop communication in SMAC as

$$E[D] = E[t_{cs} + t_{tx} + t_s] = t_{cs} + t_{tx} + T_f/2 \tag{3}$$

For DSMAC, as described above, we assume three duty cycle levels. Let T_f denote the initial SMAC frame length. Let p_1, p_2 and p_3 denote respectively the probabilities that T_f, $T_f/2$, $T_f/4$ are the adjusted duty cycles in the algorithm. We have

$$\begin{aligned} E[D] &= E[t_{cs} + t_{tx} + t_s] && (4)\\ &= t_{cs} + t_{tx} + T_f(p_1/2 + p_2/4 + p_3/8) && (5)\\ &= t_{cs} + t_{tx} + \frac{T_f(4p_1 + 2p_2 + p_3)}{8} && (6)\\ & && (7)\\ p_1 + p_2 + p_3 &= 1 && (8) \end{aligned}$$

It follows that

$$\begin{aligned} 4p_1 + 2p_2 + p_3 &< 4 \Leftrightarrow && (9)\\ \frac{4p_1 + 2p_2 + p_3}{8} &< 1/2 \Leftrightarrow && (10)\\ E[D]_{DSMAC} &< E[D]_{SMAC} && (11) \end{aligned}$$

From the above analysis, it is easy to conclude that in DSMAC, the higher the probabilities p_3 and p_2, the lower the average latency. In other words, when the measured delay constraint gets more stringent, the probability of adopting a higher duty cycle and a lower frame length is increased, thus, the average latency is decreased.

2) Multiple-hop Case: This is an extension of the single-hop case with similar analysis.

In DSMAC, assume the largest duty cycle is adopted along the path. It follows from [9] that

$$E[D(N)] = NT_f/4 - T_f/8 + t_{cs} + t_{tx} \tag{12}$$

where N is the number of hops, the average latency of DSMAC with adaptive listen is

$$E[D(N)] = NT_f/8 + 2t_{cs} + 2t_{tx} - T_f/8 \tag{13}$$

However, since the SYNC packet is sent during every SYNC period, it takes at least $(N-1)SYNCPERIOD$ for the nodes along the path to have all their duty cycles updated in the beginning of the transmission. Here, $SYNCPERIOD$ is the SYNC period length.

IV. Simulation Results

In this section, we will present the experimental results which demonstrate the tradeoffs between latency and energy. We will compare the performance of SMAC and DSMAC via simulations. The SMAC implementation we use for comparison is the version implemented in ns2-2.26.

Parameter	Value	Description
SMAC_MAX_NUM_NEIGHBORS	20	Maximum number of neighbors
SMAC_MAX_NUM_SCHEDULES	4	Maximum number of schedules
SMAC_DUTY_CYCLE	10	SMAC duty cycle in percentage
SYNCPERIOD	10	number of cycles between SYNC packets
BANDWIDTH	20	channel bandwidth (kbps)
SYNC_CW	31	SYNC packets contention window size
DATA_CW	63	data packets contention window size
Pt_consume	0.660	power consumption for transmission (W)
Pr_consume	0.395	power consumption for reception (W)
P_idle	0	power consumption for sleep (W)
P_Init	100	initial energy level (J)

TABLE I

GENERAL SIMULATION PARAMETERS

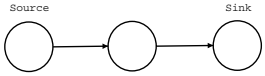

Fig. 3. Two-hop network topology

A. Simulation Configurations

The simulations presented in the section are done on ns2(ns-2.26) [5] with CMU Monarch Wireless and Mobility extensions. The general simulation parameters are given in Table I. Other scenario-specific parameters are listed in the respective scenario. In the simulations, no mobility is assumed.

For both SMAC and DSMAC, RTS/CTS handshaking mechanisms have been assumed. For DSMAC, Dmin is set to be 1 second and Dmax is set to be 2. T_E is set to be 0.1. The traffic sources are turned on after 50 seconds.

B. Power and Delay Performance

The goal of this set of simulations is to demonstrate the tradeoffs between the conflicting goals of saving energy and reducing latency. To compare the performance of SMAC and DSMAC, the average end-to-end latency and power consumption values have been measured.

In order to concentrate on the inherent properties of SMAC and DSMAC, we perform the tests on a simple two-hop network topology as illustrated in Figure 3. In this simple network, there is one source and one sink. The source is sending fixed-length packet of 512 bytes at a constant rate. We measure the average delay experienced by the receiver and power consumption for both the sender and receiver. Several source traffic loads, varying from light to heavy, have been simulated.

Figure 4 shows the average latency measured at the destination sensor node for three algorithms accordingly. The Full duty cycle algorithm(i.e., 802.11-like), denoted as "FULL" in the legend, assumes no sleep period at all, and hence achieves the best delay performance. Both SMAC and FULL have a

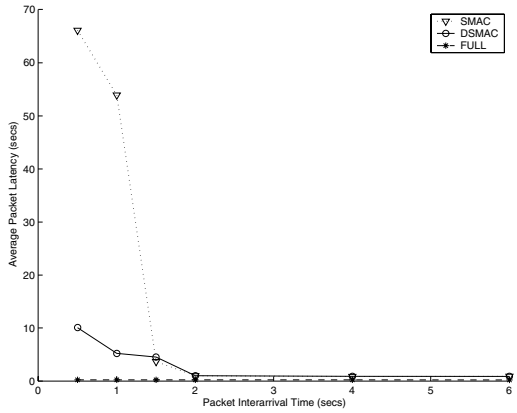

Fig. 4. Latency Behaviors of SMAC, DSMAC and 802.11-like schemes

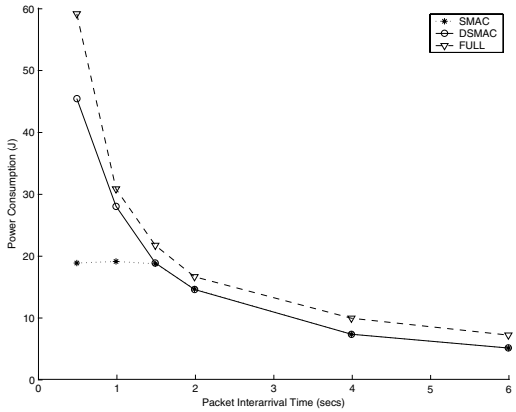

Fig. 5. Total Power Consumption of SMAC, DSMAC and 802.11-like schemes

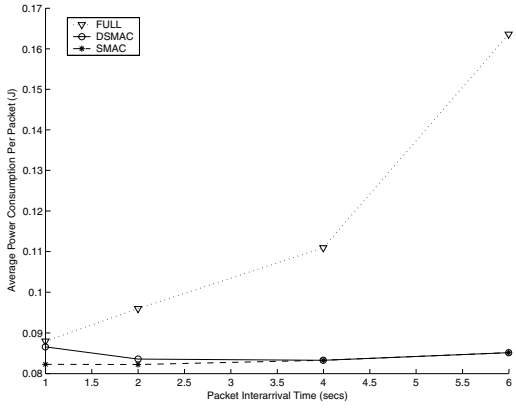

Fig. 6. Average Power Consumption Per Packet of SMAC, DSMAC and 802.11-like schemes

more smooth delay curve as they both assume a constant duty cycle once determined. However, since SMAC has a much lower duty cycle based on the assumption of low average network load, the latency of SMAC is much higher than FULL, which always utilize the full system capacity. DSMAC achieves a tradeoff between these two extremes. Once the delay experienced at the destination exceeds the threshold value, the autonomous duty cycle adjustment will be activated to keep the latency from increasing until the system resource limit has been reached.

Figure 5 shows the sum of the power consumption values measured at the sender and the receiver after the end of the simulation runs for three algorithms accordingly. Figure 6 shows the average power consumption per packet. From the result, it is clear that when traffic load gets higher, DSMAC adjusts its duty cycle to accommodate the latency demands until the maximum duty cycle level is reached. Moreover, when the traffic load is lower, DSMAC degenerates itself to SMAC while FULL is consuming more power for the same amount of packets delivered. In Figure 5, the power saving property of periodic sleeping sensor MAC has dominated when the sensor becomes inactive(i.e., a larger packet interarrival time). DSMAC has taken advantage of its adaptive duty cycle adjustment scheme to return back to intensive sleeping cycle when transmission demands become small. When the traffic volume turns higher, DSMAC performs similarly as FULL. Both FULL and DSMAC are sending more packets than SMAC when the traffic load is higher. When the traffic load is increased, Figure 5 shows a gap between DSMAC/FULL and SMAC in total power consumption for all packets sent, however, Figure 6 shows that for DSMAC and FULL, the actual per packet power consumption efficiencies are improving. Figure 6 shows that DSMAC achieves high power efficiency when the traffic load is low while being able to adjust itself to accommodate the high traffic load to improve the latency performance.

C. Coexistence of Multiple Duty Cycles

In DSMAC, different neighboring sensor nodes may take different duty cycles even though they can hear each other. Sensor nodes with higher traffic load are allowed to increase their duty cycles. However, most of the time, neighboring nodes can still stick to their own low duty cycle if they are lightly loaded and perceive a much better latency. This property is important because for nodes without much activity, the higher duty cycles, the more energy waste.

Table II demonstrates the feasibility of the coexistence of multiple duty cycles in a local region. The simulation topology is shown in Figure 7. In the simulation, node 0 sends packets to node 1 and node 2 sends packets to node 3. The four sensor nodes form a line equidistantly. Nodes 0 and 2 are close to each other, however, they are still using different duty cycles most of the time, because the traffic between nodes 0 and 1 is higher than that of nodes 2 and 3. In order to examine the interaction between two coexisting traffic loads, we keep the traffic load at node 2 at a low level (interarrival time is 4 seconds) with a a high traffic load at node 0 (interarrival time is 0.5 seconds), and investigate the time duration in seconds for a node to stay at a certain duty cycle. The power consumption values are also shown in the result. The result shows that the durations for nodes 0 and 1 to stay with a high duty cycle are much longer than those for nodes 2 and 3. Node 2's high duty cycle duration is longer than that of node 3, due to its contention with node 0 when sending packets.

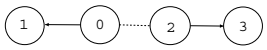

Fig. 7. Network topology for co-existing duty cycles study

	Duration 10	Duration 20	Duration 40	Power Consumption(J)
0	192.02401	94.32201	109.65500	17.67
1	179.50000	14.30000	192.48801	14.68
2	334.26302	55.69200	6.04899	12.8
3	281.35900	9.29200	7.48699	2.12

TABLE II
NEIGHBORING NODES OF DIFFERENT DUTY CYCLES

V. CONCLUSION

In this paper, a Dynamic Sensor MAC protocol with an adaptive duty cycle called DSMAC has been proposed, in which the tradeoff between power consumption and latency has been studied. The tradeoff between energy and latency is an important issue in the sensor networks. Due to the limitation in sensor battery supply, it is not efficient to keep sensors active over all time. On the other hand, some sensor applications may desire a small latency. For periodic-sleep MAC protocols, the issue of synchronization among neighbors is also important and has influenced the sensor MAC designs significantly. The proposed DSMAC supports multiple duty cycles automatically adjusted, based on the measurement of the energy consumption level and delay. Both analysis and simulations have shown an improved latency performance with the energy consumption at a reasonable level.

REFERENCES

[1] Ian F. Akylidiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Personal Communications Magazine*, August 2002.

[2] Deepak Bansal, Manish Kalia, Dinesh Anvekar, and Rajeev Shorey. MAC scheduling and SAR policies for bluetooth: A master driven TDD pico-cellular wireless system. *MoMuc*, Nov 1999.

[3] K. Sohrabi et al. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications Magazine*, Oct 2000.

[4] Jason Hill. *System Architecture for Wireless Sensor Networks*. PhD thesis, UNIVERISY OF CALIFORNIA at BERKELEY, 2003.

[5] The VINT Project. The ns manual. http://www.isi.edu/nsnam/ns/doc/index.html.

[6] the working group for WLAN standards. IEEE 802.11 standards, Part 11: Wireless Medium Access Control(MAC) and physical layer(PHY) specifications. Technical report, IEEE, 1999.

[7] A. Woo and D. Culler. A transmission control scheme for media access in sensor networks. In *Mobicom*, July 2001.

[8] Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of the IEEE INFOCOM*, June 2002.

[9] Wei Ye, John Heidemann, and Deborah Estrin. Medium access control with coordinated, adaptive sleeping for wireless sensor networks. In *IEEE Transactions on Networking*, June 2003.