Minimizing Energy Consumption in Sensor Networks Using a Wakeup Radio

Matthew J. Miller
Department of Computer Science, and
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Email: mjmille2@uiuc.edu

Nitin H. Vaidya
Department of Electrical and Computer Engineering, and
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Email: nhv@uiuc.edu

***Abstract*— For increasing the life of sensor networks, each node must conserve energy as much as possible. In this paper, we propose a protocol in which energy is conserved by amortizing the energy cost of communication over multiple packets. In addition, we allow sensors to control the amount of buffered packets since storage space is limited. To achieve this, a two-radio architecture is used which allows a sensor to "wakeup" a neighbor with a busy tone and send its packets for that destination. However, this process is expensive because all neighbors must awake and listen to the primary channel to determine who is the intended destination. Therefore, triggered wakeups on the primary channel are proposed to avoid using the more costly wakeup procedure. We present a protocol for efficiently determining how large the period for these wakeups should be such that energy consumption is minimized.**

I. Introduction

The emergence of sensor networks presents many new challenges in wireless ad hoc networks. While the precise application of sensor networks is speculative, certain characteristics are typically assumed. First, the sensors are static after initial deployment (unless placed on a mobile entity [1]). Second, energy is scarce and it is inconvenient or impossible to replenish the energy source frequently.

Because energy should be conserved, power save protocols are needed. This problem can be addressed at each layer of the network stack. Our specific focus is the MAC layer since this gives a fine-grained control to switch the wireless radio on and off. There are four major sources of energy waste at the MAC layer: packet collisions, overhearing packets intended for another destination, control packet overhead, and idle listening [2]. Radios typically have four power levels corresponding to the following states: transmitting, receiving, listening, and sleeping. Typically, the power required to listen is about the same as the power to transmit and receive. The sleep power is usually one to four orders of magnitude less. Thus, a sensor should sleep as much as possible when it is not engaged in communication. The power levels for Mica Mote sensors [3] are shown in Table I.

For brevity, we present a protocol designed for a topology where all sensors are within range of each other. In general, this protocol can be extended to the multihop case. The single hop case, however, shows important properties of the protocol. As in previous work [4], [5], [6], we assume that a second radio is available to awake neighbors. This second radio uses much less power via either a low duty cycle [7], [5] or hardware design [6]. It is assumed the second radio is only capable of transmitting a busy tone, rather than actual data. This allows a simpler, more energy efficient design. However, it introduces a problem: each busy tone must wakeup a node's entire neighborhood since the intended receiver's identifier is not encoded on the *wakeup* channel. The main contribution of this paper is selectively waking up the *primary* radio at nodes that have previously engaged in communication via rate estimation. Analytically, we derive equations to find the optimal wakeup interval to minimize the energy consumption.

TABLE I
CHARACTERISTICS OF A SENSOR RADIO

Radio State	Power Consumption (mW)
Transmit	81
Receive/Idle	30
Sleep	0.003

II. Related Work

The PAMAS protocol [4] adapts basic mechanisms of IEEE 802.11 [8] to a two-radio architecture. PAMAS allows a node to sleep to avoid overhearing a packet intended for a different destination or to avoid interfering with another node's reception by transmitting. However, unlike our work, it ignores the idle listening problem.

The PicoRadio [6] design uses a low-power wakeup channel. A MAC protocol has been designed which allows nodes to wakeup a neighbor when data needs to be sent. However, the design uses a CDMA scheme which requires each neighbor within a 2-hop range to be assigned a unique channel and discover and maintain the channel IDs for each 1-hop neighbor. Also, the channel ID is encoded in the wakeup signal, which increases the hardware complexity. Our approach could be adapted to similar hardware which uses a busy tone on the wakeup channel.

A wakeup channel is also used in [9]. The protocol is implemented from off-the-shelf hardware and tested. However,

This work is supported in part by NSF grant ANI-0125859 and a NDSEG fellowship.

the protocol is designed for systems with centralized access points or proxies and not fully distributed networks.

S-MAC [2] is a protocol developed specifically to address energy issues in sensor networks. It uses a simple scheduling scheme to allow neighbors to sleep for long periods and synchronize wakeups. In S-MAC, nodes enter sleep mode when a neighbor is transmitting and fragment long packets to avoid costly retransmissions. S-MAC is designed to save energy on a single radio architecture. While this approach does allow packets to be buffered, it provides no mechanism to communicate with the receiver on-demand. Also, S-MAC uses a fixed sleep interval regardless of traffic.

STEM [7], [5] is a two-radio architecture which achieves energy savings by having the primary radio sleep until communication is necessary while the wakeup radio periodically listens using a low duty cycle. When a node has data to send, it begins transmitting continuously on the wakeup radio long enough to guarantee that all neighbors will receive the wakeup signal. A variant of STEM [5] has been proposed that uses a busy tone, instead of encoded data, for the wakeup signal. Our protocol is similar to STEM, but achieves greater energy savings by periodically listening on the primary channel and buffering packets.

T-MAC [10] extends S-MAC by adjusting the length of time sensors are awake between sleep intervals based on communication of nearby neighbors. Thus, less energy is wasted due to idle listening when traffic is light. However, T-MAC is still limited by a one-radio architecture.

In [11], energy is also saved by adjusting to traffic. The protocol works with on-demand routing and uses 802.11's power save mechanism when a node is not engaged in sending, receiving, or forwarding data. When a node is communicating, soft-timers are used to transition the node to an idle listening mode which reduces latency and preserves throughput better than only using 802.11's power save. However, the timers do not adjust to the traffic rate, so if traffic is not frequent enough to refresh the timers, the benefits of the protocol are lost.

In [12], renewal theory is used to analyze schemes where a mobile device wakes up to receive packets from a base station with a queue of size zero or ∞. The analysis allows arbitrary probability distributions for the interarrival times of packets. Similarly, our protocol attempts to intelligently wakeup to reduce energy consumption. However, our protocol is different because it benefits from a second radio.

III. Protocol Description and Discussion

As mentioned previously, there are two channels: primary and wakeup. The primary channel is used for sending data and control packets, whereas the wakeup channel is used to wakeup neighbors. For the rest of the paper, we assume that the wakeup radio achieves low power consumption via a duty cycle. That is, a node will listen for a busy tone on the wakeup channel for τ_1 time, then sleep for τ_2 time ($\tau_1 \ll \tau_2$). The sender of a wakeup signal must transmit for $2\tau_1 + \tau_2$ time to guarantee all neighbors hear the wakeup signal. The duty cycle of the wakeup channel is defined as $\frac{\tau_1}{\tau_1+\tau_2}$. Thus, a lower duty cycle reduces idle listening energy, but increases the delay to wake a node's neighborhood. A queue threshold, L, is specified for the protocol. This threshold could be used to control delay or limit the storage usage on a sensor. For simplicity, L is expressed in packets and all data packets are the same size[1]. When the queue holds L packets, a wakeup signal must be sent so the queue size can be reduced by transmitting packets to a receiver immediately. We refer to this as a *full wakeup* because all sensors within one hop of the sender, after detecting the signal, must wakeup their primary radio and listen on the primary channel until a *filter packet* is sent (on the primary channel) to indicate which sensor's radio should remain on for reception. The other neighbors then return to sleep. To avoid costly full wakeups, a sensor estimates the rate at which it is sending data and tries to schedule a *triggered wakeup* with a receiver T seconds after its previous data transmission.

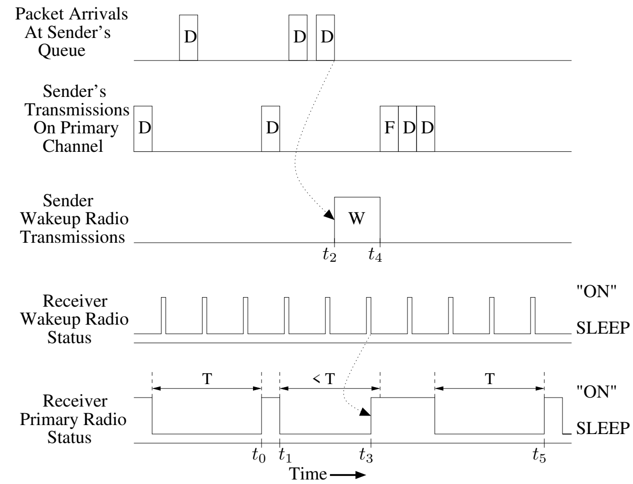


Fig. 1. Static T and $L = 2$ (**D** = data packet, **F** = filter packet, **W** = wakeup signal)

Figure 1 illustrates this concept with a fixed T value. At t_0, a triggered wakeup occurs T time after the last transmission, even though the sender's queue contains less than L packets. A full wakeup begins at t_2 because the sender's queue reaches size L. At t_4, all neighbors are guaranteed to have their primary radios on, so a filter packet (shown as **F** in the figure) and L data packets (shown as **D**) are sent on the primary channel. Unlike the figure, our protocol will dynamically adjust T since the rate is not known in advance and may vary with time.

Intuitively, if T is too small, the sender and receiver waste energy by waking up when the queue has no packets. This is called an *empty triggered wakeup*. This is shown in Figure 1 at t_5, results in idle listening because the primary radios stay on long enough, say T_{thresh} duration, to make sure no data is available (T_{thresh} is not shown in Figure 1). Thus, they are on for T_{thresh} after doing a triggered wakeup or

[1] Alternatively, L could be specified in bytes.

sending/receiving a packet. If no data is sent/received within T_{thresh} time, the primary radios return to sleep. Our goal is to find the optimal T value (T_{opt}), for a given data rate, which minimizes the energy consumption.

Initially, no triggered wakeup is scheduled and a full wakeup occurs when the queue contains L packets[2]. A timer is used to make sure a packet does not remain in the queue indefinitely if the sender stops generating packets[3]. The sender will piggyback its chosen T value (in ms) on each data packet sent. The sender and receiver will then schedule a triggered wakeup T time in the future, taking into account transmission delay. If no more data is sent or received for T_{thresh} time, the sensors will return to sleep and wakeup $T - T_{thresh}$ time later. A minimum value, T_{min}, is specified for T such that $T_{min} > T_{thresh}$. We describe how T is adjusted in Section IV-A.

From our protocol description, we see that STEM [7], [5] is a special case of our protocol with $T = \infty$ and $L = 1$. In STEM, each packet arrival causes a full wakeup, whereas our protocol avoids some full wakeups by using triggered wakeups. Our protocol is different from T-MAC [10], and similar protocols, which adjust the time a radio is on once it enters the idle state. Our protocol tries to sleep as soon as possible after data communication and predict when it should next wakeup based on previous traffic patterns.

IV. Energy Analysis of Triggered Wakeups

To find T_{opt}, we derive equations for the expected energy consumption per bit. We make some simplifying assumptions in the analysis. First, it is assumed there is one sender transmitting to one receiver among N sensors. The remaining $N-2$ nodes do not send or receive any data. Second, we assume once a sensor starts sending a wakeup signal or does a triggered wakeup, only packets in the queue *at the beginning of the wakeup* are sent. Thus, exactly L packets are sent for a full wakeup and at most $L-1$ packets are sent for a triggered wakeup. We remove this constraint in the simulations. Finally, for brevity, we leave out relatively insignificant terms, such as idle energy consumed during propagation delays and DIFS/SIFS intervals, though these terms are included in the analytical graphs. Our parameters, shown in Table I and Table II, are based on Mica Motes and 802.11. The RTS, CTS, and ACK packet sizes and contents are unmodified from the 802.11 standard. We set τ_1 and τ_2 to be 1 ms and 299 ms, respectively. These values are similar to those used in [13] and give a duty cycle of $\frac{1}{300}$. With these power levels, two radios, and the selected wakeup radio duty cycle, the average power used while sleeping, P_{sleep}, is:

$$P_{sleep} = 0.003\left(\frac{299}{300}+1\right) + 30\left(\frac{1}{300}\right) \approx 0.106 \text{ mW}$$

[2] L is not necessarily equal to the capacity of the queue.

[3] The simulated flows do not test this because packets never cease being generated.

TABLE II

Protocol parameter values

Parameter	Value
Physical Layer Header ($PLCP$)	4 bytes
Network Layer Header (IP)	20 bytes
MAC Layer Header (MAC)	32 bytes
Data Size ($DATA_{size}$)	30 bytes
Bytes in each Data Packet	$DATA_{size} + MAC + PLCP + IP$
Bytes in a Filter Packet	$33 + PLCP$
Bitrate	40 kbps
T_{thresh}	20 ms
T_{min}	50 ms

E_{bit} is the cumulative energy used by all nodes (in Joules) per data bit delivered. Recall that L is our queue threshold and N is the number of sensors. Let R be the packet arrival rate. The interarrival time of packets has an exponential distribution. Later, we consider time-varying rates. First, we derive p_f, the probability a full wakeup occurs. Let X be the length of time until the L-th packet arrival and Y be the number of packet arrivals that occur over time T (e.g., $Y \sim$ Poisson($\lambda = RT$)).

$$\begin{aligned} \Pr[X \geq T] &= \Pr[Y < L \text{ on the interval } [0, T)] \\ p_f &= \Pr[X < T] = 1 - \Pr[X \geq T] \\ &= 1 - \sum_{i=0}^{L-1} \frac{(RT)^i}{i!} e^{-RT} \end{aligned} \quad (1)$$

Let p_e be the probability of an empty triggered wakeup and $p_{\overline{f+e}}$ be the probability of a non-empty triggered wakeup. We have: $p_e = e^{-RT}$ and $p_{\overline{f+e}} = \sum_{i=1}^{L-1} \frac{(RT)^i}{i!} e^{-RT}$.

Next, let $Q_{\overline{f+e}}$ be the expected number of packets in the queue at time T for a non-empty triggered wakeup. Thus,

$$Q_{\overline{f+e}} = \frac{\sum_{i=1}^{L-1} i \frac{(RT)^i}{i!} e^{-RT}}{\sum_{i=1}^{L-1} \frac{(RT)^i}{i!} e^{-RT}} = \frac{\sum_{i=1}^{L-1} i \frac{(RT)^i}{i!}}{\sum_{i=1}^{L-1} \frac{(RT)^i}{i!}} \quad (2)$$

We need to find T_{sleep_full}, the expected sleep time given a full wakeup occurs. Let Z be the expected time of the L-th packet arrival. Thus, $T_{sleep\_full} = \text{Ex}[Z | Z \leq T]$ and $Z \sim$ Gamma($\alpha = L, \beta = \frac{1}{R}$). Therefore,

$$\begin{aligned} T_{sleep\_full} &= \frac{\int_0^T z f(z)\ dz}{\int_0^T f(z)\ dz} \\ &= \frac{\int_0^T \frac{R^L}{\Gamma(L)} z^L e^{-Rz}\ dz}{\int_0^T \frac{R^L}{\Gamma(L)} z^{L-1} e^{-Rz}\ dz} \\ &= \frac{\int_0^T z^L e^{-Rz}\ dz}{\int_0^T z^{L-1} e^{-Rz}\ dz} \end{aligned} \quad (3)$$

Now, we can express the expected energy consumed for each type of wakeup. Let

$$E_{pkt} = E_{MAC\_RX} + E_{data\_RX} + E_{MAC\_TX} + E_{data\_TX}$$

be the energy required to send *and* receive one packet, where E_{data_TX} and E_{data_RX} is the energy to send and receive a

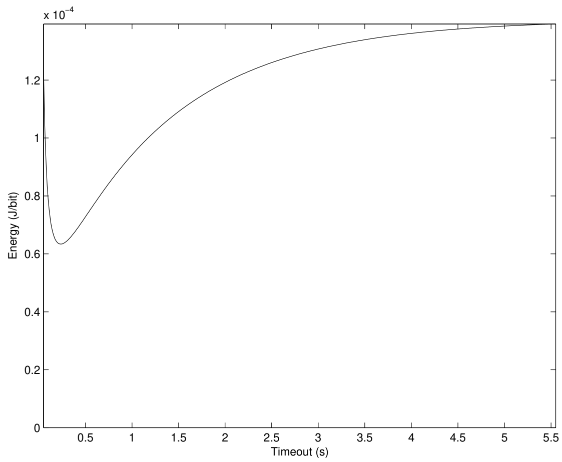


Fig. 2. Energy versus timeout

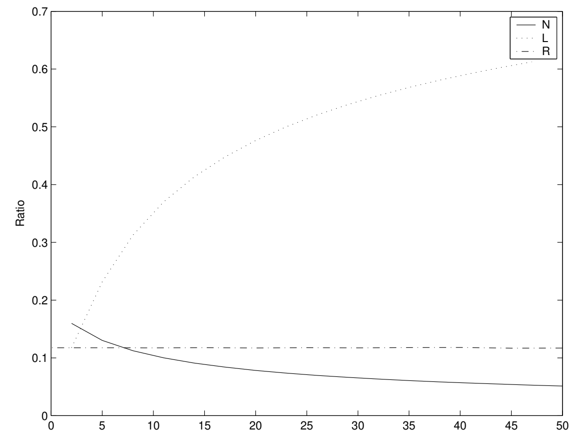


Fig. 3. $\frac{T_{opt}}{L/R}$ ratio

data packet, respectively. E_{MAC_TX} is the energy to send an RTS and receive a CTS and ACK. Similarly, E_{MAC_RX} is the energy to receive an RTS and send a CTS and ACK. For an empty triggered wake, the energy is:

$$E_{empty} = 2E_{thresh} + NP_{sleep}T \tag{4}$$

where E_{thresh} is the energy needed to listen to the channel for T_{thresh} time. For non-empty triggered wakeups:

$$E_{triggered} = Q_{\overline{f+e}}E_{pkt} + 2E_{thresh} + NP_{sleep}T \tag{5}$$

For a full wakeup, the equation is:

$$\begin{aligned} E_{full} = E_{wake\_TX} + (N-1)E_{wake\_RX} + \\ E_{filter\_TX} + (N-1)E_{filter\_RX} + \\ LE_{pkt} + 2E_{thresh} + NP_{sleep}T_{sleep\_full} \end{aligned} \tag{6}$$

Thus, the expected energy consumed per bit is:

$$E_{bit} = \frac{p_f E_{full} + p_{\overline{f+e}}E_{triggered} + p_e E_{empty}}{Data_{size} \times 8 \times (p_f L + p_{\overline{f+e}}Q_{\overline{f+e}})} \tag{7}$$

Using Equation 7, Figure 2 shows E_{bit} as a function of T for $R = 1.0$, $L = 2$, and $N = 8$. Clearly, choosing T_{opt} should minimize energy consumption.

A. Adjusting T

The sender estimates its sending rate via a weighted average of the interarrival time of packets. The estimate, $R_{est} = \frac{1}{t_{est}}$, is updated according to the equation:

$$t_{est} = \rho t_{est} + (1-\rho)t_{diff} \tag{8}$$

where t_{diff} is the most recent sample of interarrival time.

Figure 3 shows how the ratio $\frac{T_{opt}}{L/R}$ (where L/R is the expected time for the queue to reach L packets) changes with R, L, and N, based on our analysis. The horizontal axis is the value of the changing parameter (i.e., R, L, or N) while the other two parameters stay fixed. The fixed values are: $R = 1.0$, $L = 2$, $N = 8$. From Figure 3, when L and N are fixed, we observe that for some constant γ,

$$T_{opt} = \gamma \frac{L}{R} \tag{9}$$

where γ is independent of R. However, Figure 3 also shows that γ is not constant if L and N are dynamic in the network. In our evaluation, we assume L and N are known in advance. Thus, γ is calculated as a function of L and N.

V. Experimental Results

We implemented our protocol in *ns-2* [14] by modifying the 802.11 MAC and physical layers. Eight sensor nodes were placed within range of each other and a random sender and receiver were chosen to begin communicating with Poisson traffic at rate R. The remaining six nodes did not send or receive any data. We tested several R values between 0.2 and 2.0 packets per second. The resulting T_{opt} values were always greater than T_{min}. Unlike the analysis, packets were sent if they arrived after a wakeup occurred. We set $L = 2$ to demonstrate the simplest case of our protocol: rather than sending a packet immediately, we try to delay it until a triggered wakeup occurs. Each data point is averaged over ten runs and error bars show standard deviation. The simulation time was such that the expected number of packets sent was the same regardless of rate. The values in Table I and Table II were used when applicable. More extensive results are presented in [15].

First, we investigate how ρ, from Equation 8, affects energy consumption. Figure 4 shows how the energy consumption changes with ρ. Intuitively, if ρ is large, the rate estimate is slow to adjust to rate variations, but more robust to occasional outliers. From the figure, we see that energy consumption remains about the same when ρ is in the range of 0.6 to 0.975. For brevity, we omit results for larger R values, but note that the lowest energy consumption occurred within the same range of ρ. If ρ is close to 1, there is large variance since the rate estimate is primarily based on the first sample. Because the interarrival time of packets follows an exponential distribution,

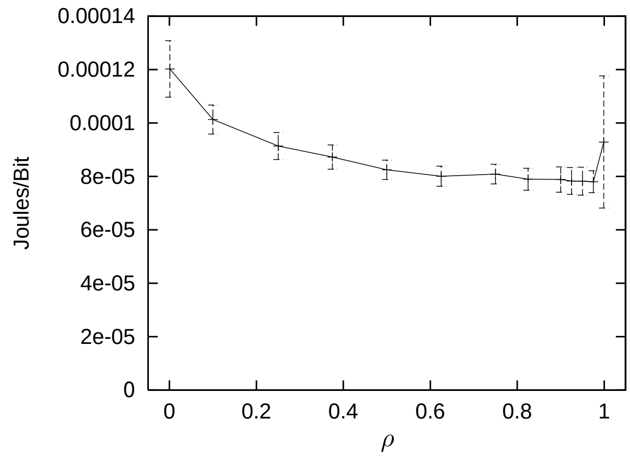
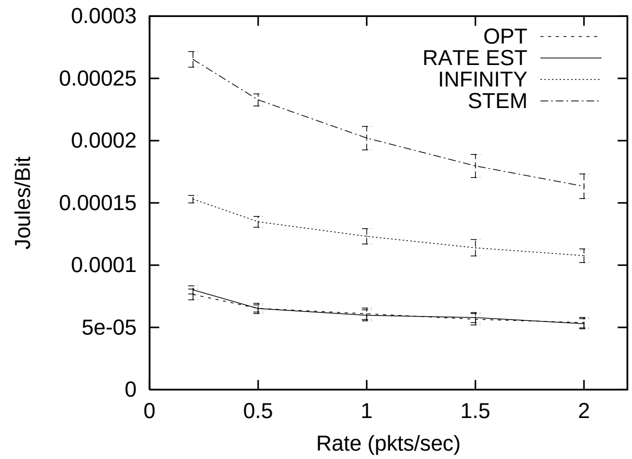
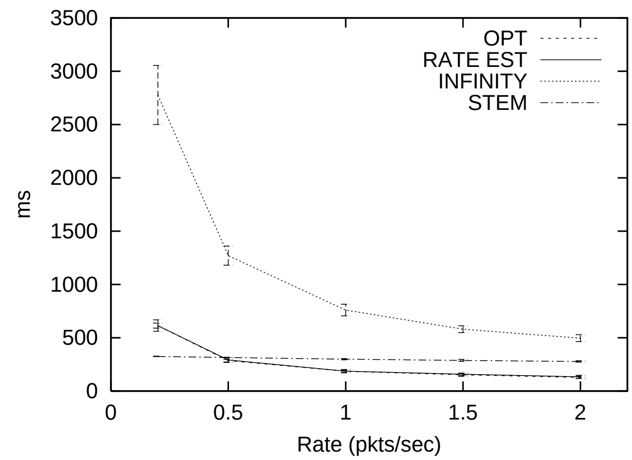


Fig. 4. Effects of ρ when $R = 0.2$



(a) Energy consumption



(b) Latency

Fig. 5. Comparison of protocols

variance is greater at a low rate (i.e., large interarrival times). If ρ is chosen to be too small, the rate estimate will not make sufficient use of previous estimates and choose T based on the most recent samples. Based on these results, we use $\rho = 0.9$ for subsequent tests, unless stated otherwise.

For comparison, we evaluated several protocols:

Rate Estimation (RATE EST) Our proposed protocol; γ is analytically calculated to be 0.1175.

Static Optimal (OPT) T is statically set to be T_{opt}, calculated analytically using the given rate. Thus, RATE EST estimates the rate dynamically, whereas OPT "magically" knows the rate.

T $= \infty$ **(INFINITY)** In this case, packets are only sent by full wakeups. Triggered wakeups never occur.

STEM This is STEM with a busy tone [5]. As mentioned in Section III, STEM is a special case of our protocol.

Figure 5(a) plots the energy consumption of the protocols with rate on the horizontal axis. This shows that regardless of rate, our protocol and the static optimal result in comparable energy consumption (the two curves almost overlap), which is significantly lower than the other protocols. As shown in Figure 5(a), the rate estimation represents about 70% improvement over STEM regardless of rate. When compared to setting $T = \infty$, the rate estimation shows about 50% improvement. This shows the need to schedule triggered wakeups even if the full wakeup cost is amortized over multiple packets.

Our protocol's performance is even better when latency is considered in Figure 5(b). Again, the rate estimation and the optimal performance overlap. Rate estimation shows more than 70% improvment compared to $T = \infty$. As expected, STEM's latency is nearly constant at each rate. At higher rates, our protocol performs better than STEM since T_{opt} is less than the time required to do a full wakeup.

A major strength of our protocol is the ability to adjust to traffic on-demand as the sending rate changes. To test the dynamic adaptation of our protocol, R was changed from 0.2 to 2.0 packets per second and back periodically. We use α to refer to the frequency with which the rate changes. More specifically, α is the expected number of packets generated at the current rate before switching to the other rate. For example, if $\alpha = 10$, packets are generated at $R = 0.2$ for $\frac{10}{0.2} = 50$ seconds, then at $R = 2.0$ for $\frac{10}{2} = 5$ seconds. This behavior was repeated for several cycles. For the static optimal, we ran two separate scenarios at the different rates and averaged the results. This essentially represents the best energy consumption possible if the protocol adjusted to rate changes immediately. Figure 6 plots energy consumption with α on the horizontal axis. For brevity, the latency results are omitted, but show trends consistent with Figure 5(b).

As expected, rate estimation does better when the sender spends a long time at a fixed rate before switching rates. When rate change is infrequent, rate estimation uses only about 5% more energy than the static optimal. When $\rho = 0.6$, rate estimation converges more quickly toward the optimal since it is more responsive to rate change. STEM and $T = \infty$ stay relatively constant and use significantly more energy than our protocol.

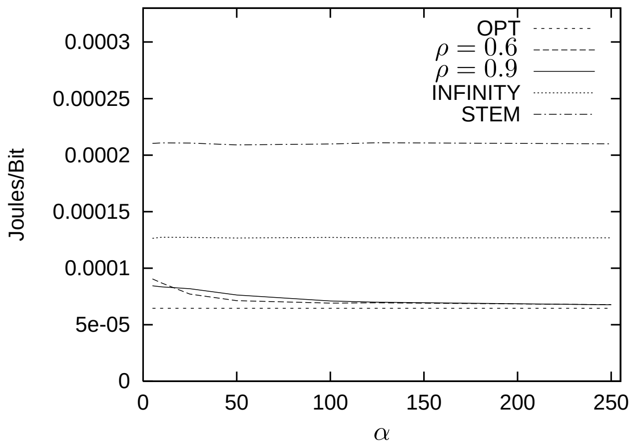


Fig. 6. Traffic with time-variant rates

VI. Conclusions and Future Work

We have analyzed a protocol for sensor networks that increases energy efficiency by allowing packet buffering, thereby amortizing the energy cost of communication over multiple packets. Because storage space may be a scarce resource in sensors, we propose adding a second, low-power radio to allow senders to force receivers to wakeup when a specified number of packets are being buffered. Our analysis reveals an optimal timeout value for periodically waking up to send and receive packets which minimizes energy consumption. Our protocol uses rate estimation to achieve results comparable to the optimal. In addition, we show significant energy savings over other, similar protocols.

In the future, we would like to adapt the protocol to work in a multihop environment. This is a challenge because each sensor would be a sender and receiver and have to adapt separate T values accordingly. Also, we would like to design a protocol to work well with multiple flows and increased contention. Finally, we would like to present a more comprehensive analysis of the latency of the protocols.

References

[1] P. Juang, H. Oki, Y. Wang, M. Martonosi, L.-S. Peh, and D. Rubenstein, “Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet,” in *ASPLOS 2002*, October 2002.

[2] W. Ye, J. Heidemann, and D. Estrin, “An Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *INFOCOM 2002*, June 2002.

[3] MICA2 Mote Datasheet, `www.xbow.com`.

[4] C. Raghavendra and S. Singh, “PAMAS – Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks,” *ACM Computer Communications Review*, July 1998.

[5] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, “Optimizing Sensor Networks in the Energy-Latency-Density Design Space,” *IEEE Transactions on Mobile Computing*, vol. 1, no. 1, pp. 70–80, January-March 2002.

[6] C. Guo, L. C. Zhong, and J. M. Rabaey, “Low Power Distributed MAC for Ad Hoc Sensor Radio Networks,” in *GlobeCom 2001*, November 2001.

[7] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, “Topology Management for Sensor Networks: Exploiting Latency and Density,” in *MOBIHOC 2002*, June 2002.

[8] IEEE 802.11, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1999.

[9] E. Shih, P. Bahl, and M. J. Sinclair, “Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices,” in *MOBICOM 2002*, September 2002.

[10] T. van Dam and K. Langendoen, “An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *SENSYS 2003*, November 2003.

[11] R. Zheng and R. Kravets, “On-demand Power Management for Ad Hoc Networks,” in *INFOCOM 2003*, April 2003.

[12] P. Nuggehalli, V. Srinivasan, K. Chebrolu, and R. Rao, “Energy Aware Sampling Schemes,” in *IEEE WCNC 2000*, September 2000.

[13] X. Yang and N. H. Vaidya, “A Wakeup Scheme for Sensor Networks: Acheiving balance between energy saving and end-to-end delay,” 2003, in submission.

[14] `ns-2` Network Simulator, `www.isi.edu/nsnam/ns`.

[15] M. J. Miller, “Minimizing Energy Consumption in Sensor Networks Using a Wakeup Radio,” Master’s thesis, University of Illinois at Urbana-Champaign, November 2003.