Exploring the Energy-Latency Trade-off for Broadcasts in Energy-Saving Sensor Networks*

Matthew J. Miller† Cigdem Sengul Indranil Gupta
Department of Computer Science
University of Illinois at Urbana-Champaign
mjmille2@uiuc.edu sengul@uiuc.edu indy@cs.uiuc.edu

Abstract

Networking protocols for multi-hop wireless sensor networks (WSNs) are required to simultaneously minimize resource usage as well as optimize performance metrics such as latency and reliability. This paper explores the energy-latency-reliability trade-off for broadcast in multi-hop WSNs, by presenting a new protocol called PBBF (Probability-Based Broadcast Forwarding). PBBF works at the MAC layer and can be integrated into any sleep scheduling protocol. For a given application-defined level of reliability for broadcasts, the energy required and latency obtained are found to be inversely related to each other. Our analysis and simulation study quantify this relationship at the reliability boundary, as well as performance numbers to be expected from a deployment. PBBF essentially offers a WSN application designer considerable flexibility in choice of desired operation points.

1 Introduction

Sensor nodes are inherently resource constrained. For example, an off-the-shelf *Mote* [1] has a lifetime of a few weeks (using a pair of standard AA batteries), short communication range distances, a 4 MHz processor, a few KBs of SRAM, and a few MBs of Flash RAM. Offering better reliability and performance to a sensor network application (e.g., tracking, environmental observation) leads to greater usage and depletion of these resources. To support a wide variety of future applications, sensor networking technologies (hardware and software) will be required to provide enough flexibility for a designer to choose the appropriate operation point on the resource-performance spectrum.

In this paper, we focus on the broadcast problem. Broadcast is useful to applications for disseminating sensor data, instructions, and code updates. We study a probabilistic approach to exploring a resource-performance trade-off for broadcast communication. Assuming an energy-conserving sensor network, our goal is to design a broadcast protocol that allows a range of operating points from which an application designer can choose. To this end, we propose PBBF (Probability-Based Broadcast Forwarding), which is a MAC-layer approach and can be integrated into any sleep scheduling protocol. While some previous studies of probabilistic broadcast in wireless networks work outside the MAC protocol [5], PBBF protocol works with the MAC protocol. We do not propose a new MAC protocol in this paper, but rather discuss a generic broadcasting protocol that can be built into any MAC layer with an appropriate sleep scheduling strategy.

To address the energy constraints of battery-powered sensors, MAC protocols use a *sleep mode*, during which little power is consumed. Examples of such protocols include S-MAC [20], T-MAC [19], and IEEE 802.11 [7]. Based on the underlying *sleep scheduling protocol*, at a given time, while some nodes are in *active mode*, others stay in sleep mode to save energy. PBBF can be added to such energy-conserving MAC protocols via two new parameters: (1) p, which is the probability that a node rebroadcasts a packet *immediately* without ensuring that any of its neighbors are active and (2) q, which is the probability that for a given node and a given time instant when it is supposed to be asleep due to its active-sleep schedule, the node instead stays awake in the expectation that it might be a receiver of an *immediate broadcast*.

Probabilistic broadcast schemes show threshold behavior; achieving a given level of reliability requires

*This research was supported in part by NSF grant ITR-0427089 and a NDSEG fellowship.
†Also affiliated with the Coordinated Science Laboratory.

the probability of forwarding to be beyond a threshold. In [5], this behavior is shown using the site percolation model. However, their approach does not allow an energy-latency trade-off. Based on our analysis using the *bond percolation* model, we show that the two knobs, p and q, introduced by the PBBF protocol can be tuned to explore the energy-latency trade-off. Essentially, only for some regions of values of p and q the threshold condition for very high reliability is satisfied, and we characterize the energy-latency trade-off primarily in this region. We find that in order to achieve a given application-defined level of reliability for broadcasts (i.e., fraction of nodes receiving the broadcast), the energy required and the latency obtained in the PBBF protocol are inversely related. While the inverse relation is not surprising, we precisely quantify the trade-off, which is essential to delineate *trade-off knobs* for the application designer.

In summary, the key contributions of this paper are (1) a new probabilistic protocol for broadcasting, (2) a precise analysis of the energy-latency trade-off allowed by these protocols for different levels of reliability, and (3) fine-grained MAC-level simulation results quantifying performance numbers for a code distribution application that use broadcasts in WSNs.

The rest of the paper is organized as follows. Section 2 discusses energy-efficient communication in WSNs. In Section 3, we describe our proposed protocol. Analytical results are presented in Section 4. Simulation results are presented in Section 5. Section 6 concludes, and presents future directions.

2 Energy-efficient Communication in Wireless Sensor Networks

In this section, we discuss various approaches for energy-efficient data dissemination in wireless sensor networks. However, these approaches mostly work outside the MAC protocol. To this end, we also present sleep scheduling mechanisms in wireless networks, which provide space for the design of an energy-efficient broadcast protocol in the MAC layer.

2.1 Efficient Broadcast Protocols

Broadcast is a fundamental communication primitive in sensor networks. Efficient broadcast techniques are essential for distributing software updates [12, 17] or sensor observations [6] among sensor nodes. The usual approach to broadcast is by flooding the entire network. This, however, creates a high number of redundant packets. While SPIN protocols [6] incorporate *negotiation* in order to avoid deficiencies of the classic flooding approach, some approaches have explored the idea of overlaying a virtual infrastructure over the underlying network [15, 16] to reduce the number of nodes involved in broadcasts. Finally, the problems with flooding can also be alleviated allowing each node to forward a message with some probability (i.e., gossip) [5, 13]. Our work in this paper is most similar to this type of approach.

It is shown that gossip-based routing [5] exhibits bimodal behavior: either virtually all or virtually none of the nodes receive the broadcast based on the gossiping probability. This problem is well-studied in percolation theory, which studies the existence of a threshold value below which infinitely many finite clusters exist and above which the cluster size approaches infinity significantly fast [3]. Similar to gossip-based routing, PBBF also affects the number of nodes that receive a broadcast since the broadcast may propagate when some nodes are in sleep mode. However, while gossip-based routing is a *site percolation problem*, where nodes broadcast with some probability [3], PBBF corresponds to a *bond percolation problem*, where bonds are open (i.e., a broadcast is sent and received) with some probability. By changing the probability a link exists in the network, PBBF provides the ability to tune the performance of an application based on the trade-off between energy, latency, and reliability.

2.2 Sleep Scheduling Mechanisms

There are two main MAC-layer approaches to reduce energy consumption in WSNs. The first approach is an active-sleep cycle, which lets nodes sleep periodically. The second approach involves using an additional low-power *wake-up radio* to wake up nodes [14]. However, since this approach requires an extra hardware component on the sensor node, the remainder of the paper focuses on only the active-sleep cycle approach.

The basic idea of introducing an active-sleep cycle to a contention-based protocol is to divide time into frames. Each frame is divided into an *active time* and a *sleep time*. During the sleep time, a node puts its radio in sleep mode to save energy. During the active time, a node can send and receive messages. For instance, the IEEE 802.11 protocol [7] provides such a power-save mode (PSM), which requires nodes to be time-synchronized and follow the same active-sleep schedule. S-MAC [20] proposes *virtual clustering* of neighbors to auto-synchronize active-sleep schedules. In both IEEE 802.11 PSM [7] and S-MAC [20], active and sleep times are fixed, while in T-MAC [19] nodes

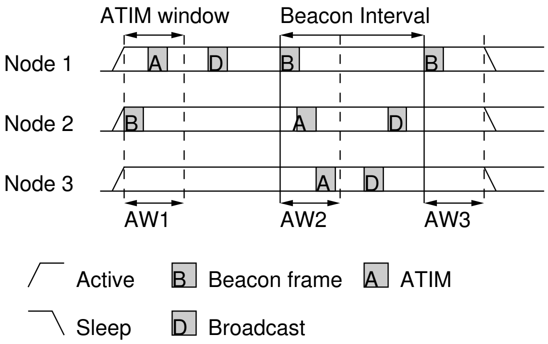

Figure 1. Broadcast in IEEE 802.11 PSM.

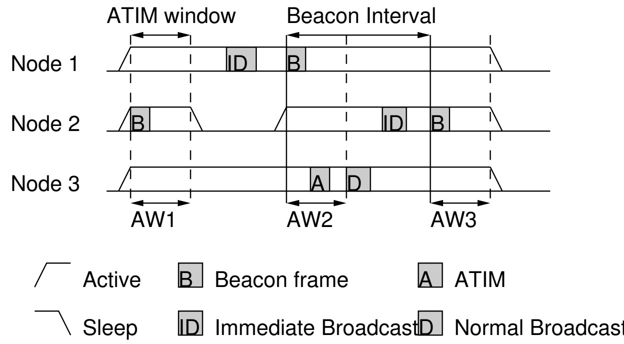

Figure 2. Broadcast in PBBF.

dynamically determine the length of active times based on communication rates.

Figure 1 shows a broadcast example for IEEE 802.11 PSM, where nodes are synchronized to wake up at the beginning of every *beacon interval*. Pending traffic is announced via ATIMs (Ad-hoc Traffic Indication Messages) in an ATIM window. In the example, Node 1 announces a broadcast ATIM for which all one-hop nodes (e.g., Node 2 and Node 3) should stay awake to receive the message after the ATIM window. An immediate observation is that to rebroadcast the message, a node must wait for the next ATIM window to guarantee that each node in its neighborhood receives the ATIM advertising the broadcast. This increases the latency of the broadcasts. A second observation is that when, say, Node 2 retransmits the broadcast message, Node 1 and Node 3 receive redundant packets. This increases energy consumption. All active-sleep scheduling mechanisms for sensor networks would display similar disadvantages. Motivated by these observations, we propose Probability-Based Broadcast Forwarding (PBBF), which allows trade-offs for latency, energy consumption, and reliability.

3 Probability-Based Broadcast Forwarding

We propose using Probability-Based Broadcast Forwarding (PBBF) that can be used in conjunction with any sleep scheduling protocol. PBBF exploits the redundancy in broadcast communication and forwards packets using a probability-based approach. Our goal is to ensure that, with high probability, a node receives at least one copy of each broadcast packet, while reducing the latency due to sleeping.

PBBF introduces two new parameters to a sleep scheduling protocol: p and q. The first parameter, p, is the probability that a node rebroadcasts a packet in the current active time despite the fact that not all neighbors may be awake to receive the broadcast. The second parameter, q, represents the probability that a node remains on after the active time when it normally would sleep.

Figure 2 shows a simple example of PBBF integrated into IEEE 802.11 PSM. In the example, Node 1 has a broadcast message to send after $AW1$. Using the p parameter, Node 1 decides to send the message immediately instead of waiting for $AW2$ to announce it. Therefore, only Node 3, which tossed a coin and decided to stay awake after $AW1$ based on the q parameter, receives the message. On reception of the message, Node 3 decides to rebroadcast via a normal broadcast and, therefore, waits for $AW2$ to guarantee that each node in its neighborhood receives the broadcast. Hence, Node 2 is able to receive the message this time and decides to rebroadcast it immediately. This example shows that if a node chooses to rebroadcast immediately, only the subset of neighbors which are currently awake can receive the packet, but with no sleep-induced delay. However, there may be no nodes to receive the packet (e.g., this would be the case if Node 3 were not awake after $AW1$ when Node 1 transmitted). The q parameter is used to avoid this problem as much as possible by allowing nodes to stay awake regardless of their active-sleep schedules.

Figure 3 shows pseudo-code of changes to any sleep scheduling protocol required for PBBF. The original sleep scheduling protocol is a special case of PBBF with $p = 0$ and $q = 0$. The *always-on* mode (i.e., no active-sleep cycles) can be approximated by setting $p = 1$ and $q = 1$. PBBF is still slightly different than *always-on* in this case because it still has the byte overhead (e.g., sending synchronization beacons) and temporal overhead (i.e., PBBF cannot send data packets during the ATIM window) of active-sleep cycles.

Through the use of two parameters, p and q, PBBF protocol provides a trade-off between energy, latency, and reliability. While p presents a trade-off between

```
SLEEP-DECISION-HANDLER()
  /* Called at the end of active time */
  /* If stayOn is true, remain on; otherwise sleep*/
stayOn ← false

if DataToSend = true or DataToRecv = true
  then
        stayOn ← true
  else if UNIFORM-RAND(0, 1) < q
          then stayOn ← true


RECEIVE-BROADCAST(pkt)
  /* Called when broadcast packet pkt is received */
if UNIFORM-RAND(0, 1) < p
  then SEND(pkt)
  else ENQUEUE(nextPktQueue, pkt)
```

Figure 3. Pseudo-code for PBBF.

latency and reliability (i.e., the fraction of nodes receiving a broadcast), q presents a trade-off in terms of energy and reliability. As p increases, latency decreases while the fraction of nodes not receiving a broadcast increases (unless $q = 1$). As q increases, energy consumption increases, but the fraction of nodes receiving a broadcast increases (unless $p = 0$). By specifying these two parameters, we investigate the energy, latency, and reliability trade-offs in the next section.

4 Analytical Results

We analyze the PBBF protocol by using a combination of theory and simulations. Simulations are required because we find a complete analysis to be intractable, in spite of several available theoretical frameworks such as percolation theory. For the simulations in this section we assume an ideal MAC and physical layer with no collisions or interference. In Section 5, we present simulation results using *ns-2* [18], which show that the general trends observed in the idealized simulations still hold when realistic MAC and physical layers are used. Additionally, although PBBF is not specific to any sleep scheduling protocol, IEEE 802.11 PSM [7] is used as the sleep scheduling protocol since it provides a complete solution for broadcast, unlike the protocols in [19, 20].

We consider a grid network topology, where each node is connected to four neighbors except the nodes on the boundary (i.e., a square lattice with no wrapping on the axes) and the broadcast source is as near to the center of the grid as possible. Table 1 lists the

Table 1. Analysis parameter values.

Parameter	Value
N	5625 (75 × 75)
P_{TX}	81 mW
P_I	30 mW
P_S	3 μW
λ	0.01 packets/s
L_1	$\approx$ 1.5 s
T_{frame}	10 s
T_{active}	1 s

parameters used in the simulation part of the analysis. N is the number of nodes in the grid, λ is the rate that broadcasts are generated at the source, and T_{active} and T_{frame} is the time nodes are active each frame and the time between frames, respectively. So, for example, when there is no traffic being advertised, each node will listen to the channel for $\frac{T_{active}}{T_{frame}}$ percent of the time. L_1 is a latency value described in Section 4.3. Its chosen value is based on empirical data observed in our simulations in Section 5. P_{TX}, P_I, P_S are the power levels of the sensor radio to transmit, receive/idle, and sleep, respectively. The values we use are based on Mica2 Motes [8].

4.1 Reliability

The reliability of PBBF protocols can be analyzed using *percolation* models. Percolation theory states that a gossip initiated by a source, n_0 dies out if there is a set of nodes, $\mathbb{N}$, that disconnects n_0 from the rest of the graph. In PBBF, $\mathbb{N}$ is the set of nodes that send an immediate broadcast which is not received by any of its neighbors (e.g., because the neighbors' coin flips result in values less than q).

Percolation theory mainly studies two percolation models: *bond percolation* and *site percolation* [3]. Let $G(V, E)$ be an infinite connected graph, where V is the set of nodes and E is the set of edges. In the bond percolation model on G, there is collection of $(X_e : e \in E)$ of independent Bernoulli random variables, each with the same mean, p_{edge}, corresponding to the set E of edges (or "bonds"). If $X_e = 1$, then the edge is *open*; otherwise it is *closed*. Given any two nodes, x and y, x can reach y (i.e, $x \leftrightarrow y$), if there exists a path of open edges between x and y. The set of nodes, which can be reached by a specific node n_0 (e.g., the source of the broadcast) is denoted by C_0, where:

$$C_0 = \{x \in V : n_0 \leftrightarrow x\}. \tag{1}$$

Percolation theory calculates conditions under which C_0 is infinite, in other words, the values of p_{edge} for which the probability $\theta^{bond}(p_{edge})$ of the component C_0 being of infinite size, is close to 1.

The *bond critical probability* $p_c^{bond}(G)$ is defined as:

$$p_c^{bond}(G) = sup\{p_{edge} : \theta^{bond}(p_{edge}) = 0\}, \qquad (2)$$

so that $\theta^{bond}(p_{edge}) = 0$ if $p_{edge} < p_c^{bond}(G)$.

The site percolation model differs because, instead of cutting given edges (bonds) in the graph with some probability, each node (site) in the graph is subjected to removal with some probability. This corresponds to the analysis of the gossip-based routing protocol in [5] where each node decides probabilistically whether to broadcast to either all its neighbors or none of them.

PBBF's reliability is characterized by a bond percolation model. First, if a node A receives the broadcast message, the probability that a given neighbor, B, of A receives a copy of the message via the link $A \rightarrow B$ is $p \cdot q + (1 - p)$. The first term arises from the likelihood of A broadcasting the message immediately after reception *and* that B being awake at the time. The second term is simply the likelihood of a rebroadcast when B is awake (i.e., the beginning of next active time). Then, each (directed) edge in the network is *open* with this probability. It must be noted that even though we assume symmetric links, a broadcast traverses a link only once, since nodes drop a broadcast packet if they receive a duplicate. Hence, by associating each (directed) edge in the network with a probability $p_{edge} = 1 - p \cdot (1 - q)$ of being present, we can say the following [3]:

Remark 1 (p and q for high reliability:) *If $p_{edge} = 1 - p \cdot (1 - q) \geq p_c^{bond}(G)$, the broadcast is received at infinitely many nodes.*

We now show PBBF's reliability using our ideal simulator by varying q while keeping p fixed. For each level of reliability (e.g., 90% and 99%), threshold behavior is observed as shown in Figure 4 and Figure 5. For sufficiently large values of p, none of the broadcasts achieve a desired reliability when q is small. However, at some threshold q value, the reliability metric rapidly increases to where every broadcast is received by the specified fraction of nodes. This is similar to the critical probability behavior shown in percolation theory [3].

We use a fast Monte Carlo algorithm from [9] to investigate the critical bond ratio for different reliability measures in grid networks (see Figure 6). We observe that, for a higher level of reliability, as expected, a larger number of bonds are required to be present. The p and q values necessary to achieve various levels of reliability in 30×30 grid network are shown in Figure 7. Each point on the figure is obtained by calculating p_{edge} from values of p and q just enough to

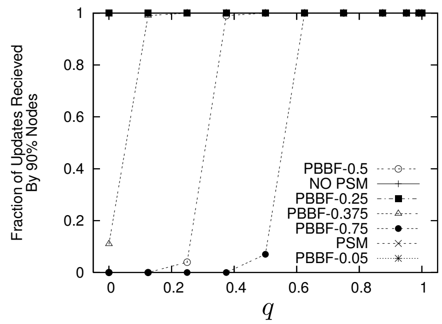

Figure 4. Threshold behavior for 90% reliability.

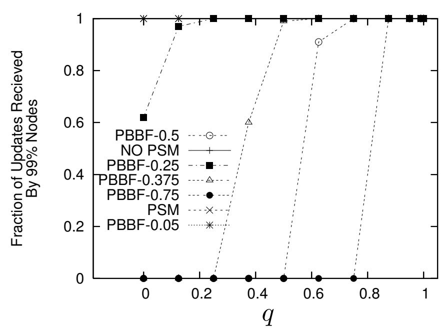

Figure 5. Threshold behavior for 99% reliability.

achieve the p_c^{bond} for a 30×30 grid network. Hence, for a given p, the parameter q should be selected from the region above the line corresponding to a reliability level. These results show the direct relationship between p and q for a given level of reliability.

4.2 Energy

Assuming the underlying sleep scheduling protocol divides time into frames and denoting active time as T_{active} and sleep time as T_{sleep}, relative energy consumption of a sleep scheduling protocol compared to a protocol with no energy-saving, $E_{original}$, can be written as:

$$E_{original} = \frac{T_{active}}{T_{frame}} \qquad (3)$$

where

$$T_{frame} = T_{active} + T_{sleep} \qquad (4)$$

The PBBF protocol allows nodes to stay active, regardless of their active-sleep schedules, based on the q parameter. Therefore, the new active and sleep times

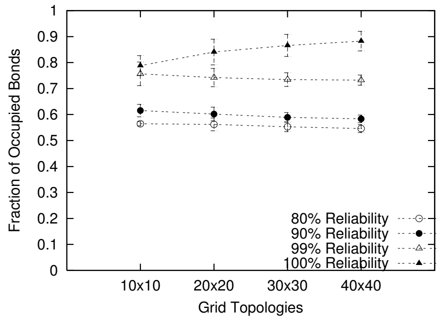

Figure 6. p_c^{bond} for various grid sizes

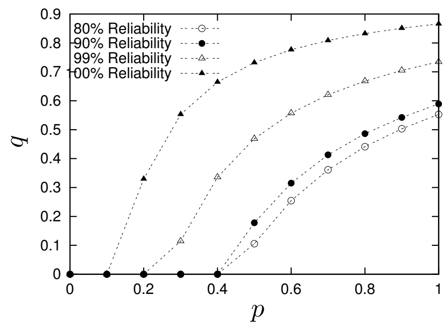

Figure 7. Relationship between p and q for a given reliability level in a 30×30 grid network.

in PBBF, $T_{active:PBBF}$ and $T_{sleep:PBBF}$, are:

$$T_{active:PBBF} = T_{active} + q \cdot T_{sleep} \quad (5)$$
$$T_{sleep:PBBF} = (1 - q) \cdot T_{sleep} \quad (6)$$

The relative energy consumption of PBBF, E_{PBBF}, is:

$$E_{PBBF} = \frac{T_{active:PBBF}}{T_{frame}} = \frac{T_{active} + q \cdot T_{sleep}}{T_{frame}} \quad (7)$$

The increased energy consumption due to the q parameter compared to original sleep scheduling protocol is:

$$\frac{E_{PBBF}}{E_{original}} = \frac{T_{active} + q \cdot T_{sleep}}{T_{active}} = 1 + q \cdot \frac{T_{sleep}}{T_{active}} \quad (8)$$

Although T_{active} and T_{sleep} are assumed to be fixed in Equation 8, these parameters can also be variables of a probabilistic distribution. The simulation results verify the analytical result given in Equation 8 (see Figure 8). While using PSM saves almost 3 Joules per update over using no PSM, the energy consumption of PBBF increases linearly with the q parameter, and does not depend on p at all (the lines for different values of p overlap).

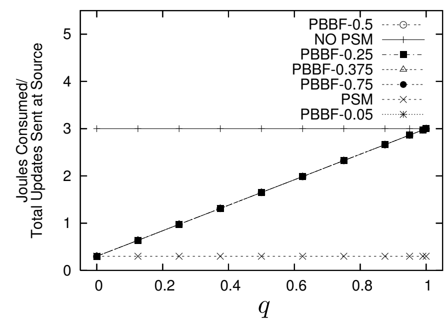

Figure 8. Average energy consumption.

4.3 Latency

For a given node, A, and a neighbor of A, B, we calculate the expected time, L, between A sending the broadcast and B receiving it from A (assuming a successful transmission from A to B).

The probability that the broadcast is sent and received immediately is $p \cdot q$, the product of the probability of an immediate broadcast (p) and that node B stays awake (q). The probability of the broadcast being sent with the assurance that all nodes wake up is simply $(1 - p)$. Thus, if the time to immediately transmit the data packet is denoted as L_1 and the time to wake up all neighbors for the broadcast is L_2, then L can be calculated as:

$$\begin{aligned} L &= \frac{L_1 \cdot p \cdot q + (L_1 + L_2) \cdot (1 - p)}{p \cdot q + (1 - p)} \\ &= L_1 + L_2 \cdot \frac{1 - p}{1 - p + p \cdot q} \end{aligned} \quad (9)$$

It must be noted while L_1 is determined by the MAC protocol (i.e., the channel access time), L_2 depends on how the sleep scheduling mechanism handles broadcast communication (i.e., ensures all nodes receive the broadcast packet). L_1 and L_2 can be either constants or variables of a probabilistic distribution.

When calculating the overall latency from the source, we need to account for the fact that a broadcast can potentially traverse through multiple different paths from the source node to a given node, B. In other words, the actual latency from the source to the node B is a function of L and the average length (in terms of hop count) $len(S, B)$ of the path from the source node, S, to node B:

$$L_{S,B} = L \cdot len(S, B) \quad (10)$$

$len(S, B)$ may be greater than the shortest distance from the source to node B since links exist on the graph based on p_{edge}. Specifically, assuming a grid network,

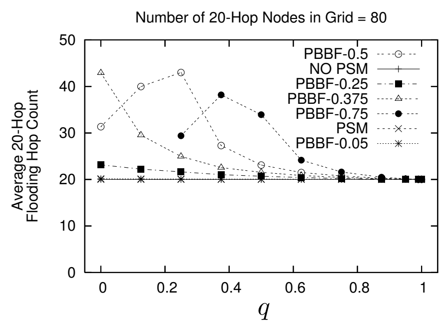

Figure 9. Average hops traveled by an update to reach a node 20 hops from the source.

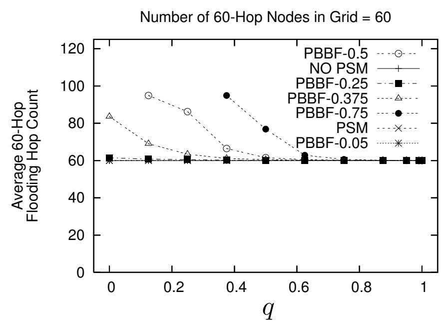

Figure 10. Average hops traveled by an update to reach a node 60 hops from the source.

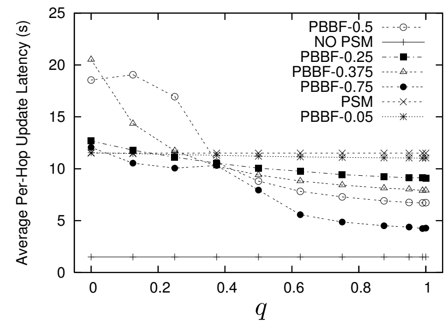

Figure 11. Average per-hop update latency.

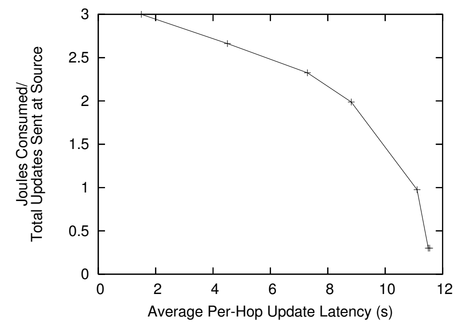

Figure 12. Energy-latency trade-off for 99% reliability.

when the source broadcasts a packet, the packet starts propagating in the network in four directions. Since nodes that receive a duplicate do not rebroadcast the packet, each broadcast message builds a uniform spanning tree. It has been shown that on such a spanning tree, the expected number of vertices on the arc from the source that lie within distance d is $d^{5/4+o(1)}$ [4,10]. From this, we can upper bound the average latency for a broadcast to reach a given node at a shortest distance, d, from the source as follows:

$$L_{S,B} \leq L \cdot d^{5/4+o(1)} \tag{11}$$

where d is the distance between S and B. However, from Figure 9 and Figure 10, we observe that when the reliability is high (points toward the righthand side of the plots), the latency $L_{S,B}$ is indeed proportional to d, and not to a quantity as high as $d^{5/4+o(1)}$, as the the reliability approaches to 100%.

A variation of per-hop latency versus q is shown in Figure 11. Since only nodes that receive at least one update are included in this latency calculation, at small values of q, the lower latency observed by higher p values is misleading. However, as q increases (i.e., broadcasts reach more nodes), higher p values (e.g., $p = 0.75$) achieve lower latency as nodes do not incur wake-up latency (i.e., L_2) due to sleep-scheduling protocol.

4.4 Energy-Latency Trade-off

From Equation 8 and Equation 9, we can derive the direct relation between energy, E_{PBBF}, and latency, L, as:

$$E_{PBBF} = (1 - \frac{L_2 + L_1 - L}{L - L_1} \cdot \frac{1-p}{p} \cdot \frac{T_{sleep}}{T_{active}}) \cdot E_{original} \tag{12}$$

Equation 8 shows that the energy consumed at a node increases linearly with q. Equation 9 shows that the latency is inversely related to q (and also p). Thus, the energy and latency are inversely related to each other in the PBBF protocol. Determining the minimum value of q for a given value of p that gives 99% reliability (see Figure 5), the energy-latency trade-off with 99% reliability is illustrated in Figure 12.

In summary, the threshold behavior of the PBBF protocols allows an application designer to first set the values of p and q so that they are just across the reliability threshold boundary and into the high reliability

Table 2. Code distribution parameter values.

Parameter	Value
N	50
q	0.25
Δ	10.0
Total Packet Size	64 bytes
Data Packet Payload	30 bytes

region. Secondly, it allows application designers to tune these values (staying close to the boundary) until the desired energy-latency trade-off is achieved.

5 Simulation Results

The goal of our simulation study is to measure our success in meeting the design goals of PBBF and investigate the trade-off between energy, latency, and the percentage of nodes receiving a broadcast. We also do the simulations to verify that the trends from Section 4 hold when collisions and interference are present. We implemented PBBF on top of IEEE 802.11 PSM and compare it with regular IEEE 802.11 PSM using the *ns-2* [18] network simulator. In addition to these MAC and physical layer modifications, we implemented an application to simulate code distribution in a sensor network.

Our implementation does not handle synchronization of nodes. Because PSM's time synchronization mechanism is only designed for single-hop networks and synchronization in multi-hop networks is a hard problem for which no good solutions currently exist [2], we assume perfect synchronization in the network. This is an assumption that other MAC protocols for sensors have made as well (e.g., [11]). The length of the beacon interval, BI, and ATIM window, AW, are set according to the values of T_{frame} and T_{active}, respectively, in Table 1. The power levels are also based on those shown in Table 1. The bit rate of the nodes is 19.2 kbps. Other parameters used in the simulations are shown in Table 2. The Δ parameter represents node density and is explained below. When applicable, the values of our parameters are based on Mica2 Mote hardware [8].

5.1 Code Distribution Application

We implemented the broadcast application at the routing layer of *ns-2*. The protocol is relatively simple. One random node is chosen to be the broadcast and code distribution source for each scenario. Each broadcast packet contains the k most recent updates generated at the source. Thus, nodes do not need to receive every broadcast as long as they receive about $\frac{1}{k}$-th of the packets. The k parameter represents a trade-off in byte overhead versus the number of packets missed by a node.

For update generation, new updates are generated and sent deterministically at the source at a rate of λ updates/second. We use the value of λ shown in Table 1. The total size and data payload of each packet are the same for all packets. The simulation values for these parameters are shown in Table 2.

To test PBBF in this setting, we varied the p and q values as well as the network density, Δ. To define Δ, we use the following equation:

$$\Delta = \frac{\pi R^2 N}{A} \tag{13}$$

where R is the range of a node, N is the number of nodes, and A is the area of the region where nodes are located. For our simulations, we fixed N and changed A to get the desired Δ. The fixed value of N is shown in Table 2. Also, in Table 2, the q and Δ parameters show the fixed value when that particular parameter is not changed. For example, when q is being varied on the x-axis, Δ is fixed at the values in Table 2. We also experimented with different values of k, but for space considerations, we only present experiments where $k = 1$ in this paper. We ran each simulation for 500 seconds and each data point is averaged over ten runs.

5.2 The impact of the q parameter

Our first experiments show how various values of q affect PBBF. Recall that q affects how likely it is that a node is able to receive an immediate broadcast. Figure 13 shows how the average energy consumed at a node, normalized for the number of updates generated, changes with q. We can see that using PSM saves almost 2 Joules per update over using no PSM. The figure also shows that energy increases linearly with the q value. We also observe that q dominates p in the energy usage because regardless of the p value, the PBBF lines overlap.

In Figure 14 and Figure 15, we see the impact of q on latency. Figure 14 and Figure 15 show the average latency of nodes that are two hops and five hops from the source, respectively. In our simulations, new packets always arrive at the source during the ATIM window, so they are sent with a delay of about AW. As expected, the latency to reach two hop neighbors is about $AW + BI$. We can see that PSM consistently has a high latency, whereas turning PSM off results in a much lower latency. PBBF does worse than PSM at small values of q, but improves significantly as q and p increase. The reason PBBF performs worse for small

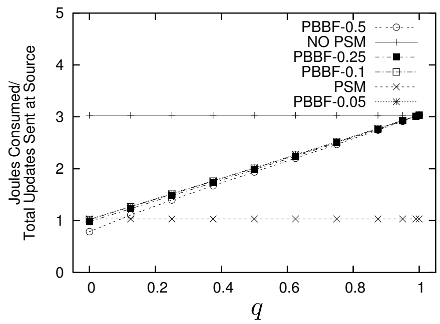

Figure 13. Average energy consumption.

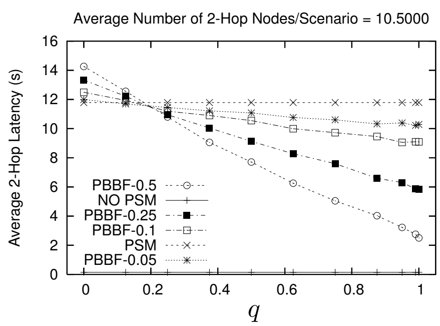

Figure 14. 2-hop average update latency.

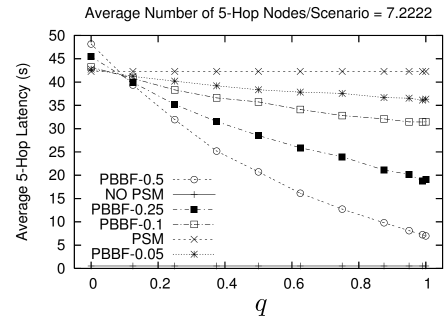

Figure 15. 5-hop average update latency.

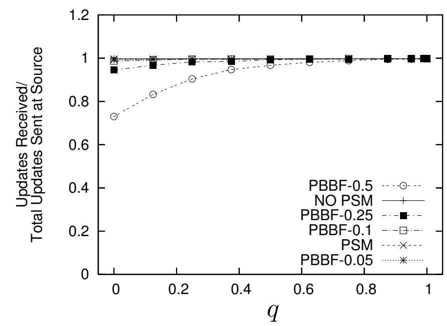

Figure 16. Average updates received.

values of q is the amount of redundancy in broadcasts received from different neighbors is reduced. Therefore, it is more likely that a node will not receive the broadcast from the neighbor which would result in the smallest latency. However, as q and p get larger, there is a greater chance a broadcast will be transmitted and received without waiting for the next beacon interval. From Figure 14 and Figure 15, we can also see that the cross-over q point where PBBF does better than PSM occurs at a lower value for nodes farther from the source. This is expected since there is a greater probability that at least one node between the source and a distant node will be able to reduce the latency by a beacon interval. Also, there are potentially many more different paths by which the broadcast can reach distant nodes.

Figure 16 illustrates how the q value affects the fraction of updates a node receives. We observe that setting $p = 0.5$ results in a significant degradation until q reaches about 0.5. For $p = 0.25$, there is a little degradation and all the other p values result in less than 1% loss.

5.3 The impact of Δ

Next, we investigate how the network density affects the protocols. The node density, Δ, is approximately equal to the expected number of one-hop neighbors for a node. Figure 17 shows that latency improves as Δ increases since nodes are expected to be fewer hops from the source. The effect is most drastic on PSM and PBBF since nodes wait less beacon intervals before receiving an update. The effect on PSM and PBBF appears to be about the same with neither showing much improvement relative to the other as Δ changes. Figure 18 illustrates that PBBF does better with respect to the number of updates received as Δ increases. This is intuitive since increasing Δ increases the number of redundant broadcasts that a node receives. We omit the the energy consumption graphs for brevity but note it is relatively constant regardless of the Δ value.

6 Conclusion

We have presented, analyzed, simulated, and measured the performance of a class of probabilistic broadcast protocols for multi-hop WSNs. We have quantified the energy-latency trade-off required to obtain a given level of reliability using PBBF. This is facili-

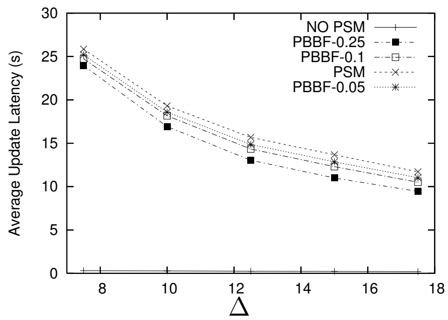

Figure 17. Average update latency.

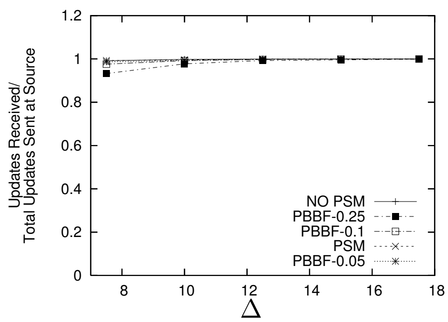

Figure 18. Average updates received.

tated by allowing an application designer to tune the values of parameters p and q while maintaining the value of $1 - p \cdot (1 - q)$ above the threshold required to achieve very high reliability. We have implemented the PBBF protocols in *ns-2*. Through simulations, we have studied the performance characteristics of PBBF when used for code distribution, an example WSN application. Our experiments indicate that PBBF is an efficient broadcast mechanism. PBBF provides an application designer the opportunity to tune the system to an appropriate operating point along the reliability-resource-performance spectrum.

In the future, we plan to explore how PBBF can be augmented to improve performance. Specifically, the p and q parameters could be adjusted dynamically by nodes. For example, when a node overhears more nodes involved in communication, p could be increased since more nodes will be active to receive the broadcast. Additionally, the q parameter could be increased in response to a node detecting a large fraction of broadcast packets are not being received. Future work will investigate these and other heuristics and in what settings p and q converge to steady-state values. Other worthwhile extensions to this work include comparing its performance with other adaptive sleep protocols and integrating PBBF with unicast power save protocols.

References

[1] Crossbow Technology. `http://www.xbow.com`.

[2] J. Elson and K. Römer. Wireless Sensor Networks: A New Regime for Time Synchronization. In *ACM Hot Topics in Networks 2002*, October 2002.

[3] G. Grimmet and A. M. Stacey. Critical probabilities for site and bond percolation models. *Annals of Probability*, 26(4):1788–1812, 1998.

[4] A. J. Guttmann and R. J. Bursill. Critical exponent for the loop erased self-avoiding walks by monte carlo methods. *Journal of Statistical Physics*, 59, 1990.

[5] Z. J. Haas, J. Y. Halpern, and L. Li. Gossip-Based Ad Hoc Routing. In *IEEE Infocom*, June 2002.

[6] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information in Wireless Sensor Networks. In *ACM MobiCom 1999*, August 1999.

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

[8] MICA2 Mote Datasheet. `http://www.xbow.com`.

[9] M. E. J. Newman and R. M. Ziff. A fast monte carlo algorithm for site or bond percolation. Technical report, Santa Fe Institute, 2001.

[10] R. Kenyon. Loop-erased random walks, Universite de Paris-Sud.

[11] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves. Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks. In *ACM SenSys 2003*, November 2003.

[12] N. Reijers and K. Langendoen. Efficient Code Distribution in Wireless Sensor Networks. In *ACM WSNA 2003*, September 2003.

[13] Y. Sasson, D. Carin, and A. Schiper. Probabilistic Broadcast for Flooding in Wireless Mobile Ad Hoc Networks. In *IEEE Wireless Communications of Networking Conference (WCNC)*, March 2003.

[14] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Topology Management for Sensor Networks: Exploiting Latency and Density. In *ACM MobiHoc 2002*, June 2002.

[15] P. Sinha, R. Sivakumar, and V. Bharghavan. Enhancing ad hoc routing with dynamic virtual infrastructures. In *IEEE Infocom*, April 2001.

[16] R. Sivakumar, P. Sinha, and V. Bharghavan. CEDAR: A core-extraction distributed ad hoc routing algorithm. In *IEEE Infocom*, March 1999.

[17] T. Stathopoulos, J. Heidemann, and D. Estrin. A remote code update mechanism for wireless sensor networks. Technical Report CENS-TR-30, UCLA, Center for Embedded Networked Computing, 2003.

[18] `ns-2` – The Network Simulator. `http://www.isi.edu/nsnam/ns`.

[19] T. van Dam and K. Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *ACM SenSys 2003*, November 2003.

[20] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *IEEE Infocom*, June 2002.