Performance Tradeoffs Among Percolation-Based Broadcast Protocols in Wireless Sensor Networks[†]

Vijay Raman
Dept. of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, IL
Email: vraman3@illinois.edu

Indranil Gupta
Dept. of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL
Email: indy@cs.illinois.edu

Abstract

Broadcast of information in wireless sensor networks is an important operation, e.g., for code updates, queries, membership information, etc. In this paper, we analyze and experimentally compare the performance of vanilla versions of several well-known broadcast mechanisms namely, flooding, site percolation, bond percolation, and modified bond percolation. We carry out our comparison for different network topologies: random, grid, and clustered. Our analysis is performed at the link layer level using a MAC-independent propagation model based on real experiments from the literature. Our main metrics are bandwidth, energy usage, and broadcast latency. Our analytical and experimental results show that, given a desired high reliability for all topologies, flooding and site percolation has the lowest latency; but flooding consumes the most energy per broadcast compared to site percolation. For dense networks, modified bond percolation further lowers energy consumption compared to site percolation, while basic bond percolation leads to a latency increase. For sparse networks, results are similar to a dense network except that site percolation consumes lower energy than modified bond percolation. We briefly discuss implications for different broadcast applications.

1. Introduction

In wireless sensor networks, operations such as code updates, queries, membership management, routing updates, etc., all rely on broadcast. Several mechanisms have been designed in the literature to address the broadcast problem [1]–[4]. In this paper, we abstract out the basic broadcast mechanisms from these papers into four classes: flooding, site percolation, bond percolation, and modified bond percolation. We analytically and experimentally compare these four approaches to each other. Although flooding has been individually compared to some of the percolation approaches [1], [4], there is a lack of an all-to-all comparison among these approaches, and thus the tradeoffs among them have not been comprehensively studied.

Broadcast is useful for varied purposes: Trickle [3] uses it for sending out continuous code updates. Aggregation using either TAG [5], directed diffusion [6], or synopsis diffusion [7] partly rely on broadcast, for example, to initiate the aggregation epochs. Routing protocols utilize broadcasts to spread routing updates. For example, [8]–[10] provides a detailed survey of various routing protocols for sensor networks. Additionally, [11] provides a survey of various energy efficient routing protocols. Membership management within small or large groups of sensor nodes has also been looked at by researchers, and utilizes broadcast, e.g., [12]. As we discuss in Section 4, these applications have highly varied requirements, and thus a one-size-fits-all broadcast solution may not work for all. This motivates our study of the tradeoffs among different broadcasting approaches.

The remainder of the paper is organized as follows: in Section 2, we briefly discuss the various broadcast mechanisms that are the focus of this paper, and derive their energy consumption. Section 3 presents simulation results of these broadcast schemes for three different network layouts. Section 4 discusses the implications of our results for practical broadcast operations. Finally, we conclude our paper in Section 5.

2. Broadcast Schemes and Energy Analysis

In this section, for each of the broadcast algorithms considered, we (1) briefly describe the algorithm, and then (2) mathematically analyze the energy consumption of these techniques.

2.1. Assumptions for the Analysis

For the mathematical analysis of the broadcast mechanisms, we make the following assumptions:

1. Every sensor node in the network is assumed to have a globally unique identity through which it can be uniquely

[†]*This work was supported in part by NSF CAREER grant CNS 04-48246 and in part by NSF ITR grant CMS 04-27089 and in part by NSF grant CNS 06-27074*

addressed. If nodes do not have unique pre-assigned identities, then the ID can be assigned either based on their location [13]–[15] or randomly.
2. The sensor nodes are static, and do not move.
3. All sensor nodes are homogeneous with respect to their transmit, receive, and idle power consumption characteristics, as well as wireless bandwidth.
4. We only worry about failure in transmissions that can occur due to the propagation characteristics of the medium so as to obtain a best estimate of the metrics. We, therefore, do not model collisions in this paper. We will discuss why this is a valid assumption in Section 4.2. Furthermore, we assume a MAC layer that takes care of retransmissions and contentions.
5. The network of sensor nodes is assumed to be connected. In other words, we assume any pair of non-faulty nodes can communicate via at least one path consisting of all non-faulty nodes. We consider only crash-recovery failures, which includes sleep schedules.

We have used a realistic propagation model that is independent of the MAC layer as done in [16]. Instead, the propagation model uses the link layer characteristics. We choose to not emulate the MAC layer because sensor nodes may use a wide variety of MAC protocols (e.g., B-MAC [17], S-MAC [17], T-MAC [18], etc.) and we wish our results to apply generically. This choice allows us to focus on incorporating loss due to realistic propagation models such as [16]. A further evaluation with individual MAC protocols could be a fruitful future work. However, as we discuss at the end of the paper, our results are likely to hold for specific MAC protocols too.

We denote by N the number of nodes in the sensor network, and by k the average number of (1-hop) neighbors of a node in the network. We analyze only the power consumed by a broadcast packet, which can be interpreted as the energy consumed by a normalized packet that takes 1 second to transmit on a link. Larger broadcast packets will consume proportionally more energy.

2.2. Flooding

This is the simplest of the three mechanisms. Each node that receives a message simply re-broadcasts it to all its 1-hop neighbors. Further, the node keeps track of recently received broadcasts, and does not re-broadcast duplicates. Thus, each node re-broadcasts a received message only once.
Analysis: In flooding, all the nodes broadcast message to their neighbors once, and the number of messages per broadcast is just 1. Therefore, the total number of messages sent per node, $S_{flooding}$ is 1. Further, since each node has k neighbors, the average number of messages received per node $R_{flooding}$ is k.

To calculate the total energy consumed, we use real measurements for Mica (2001) motes from [19], which showed that the transmit power required for these type of motes for a packet (36 mW at 0 dBm) is about thrice the power required for receiving a packet (12 mW). Furthermore, because the idle listening power is the same for all nodes (which is 8 mW) we ignore that in our calculations. Accordingly, if the power required by a sensor node for receiving a packet is denoted by Pow, the average power consumed (or energy consumed in one second) by the sensor nodes for both receiving and transmitting during the broadcast process $P_{flooding}$ is given by,

$$P_{flooding} = \frac{(3N + kN) \cdot Pow}{N} = (3 + k) \cdot Pow.$$

2.3. Site Percolation-based Gossiping

In this scheme, each node that receives a message broadcasts the message to its neighbors with a probability p, called *gossip probability*. A node does not broadcast the message with a probability $(1 - p)$. Thus, only a fraction of all nodes forward the broadcast message, thereby reducing the number of messages in the network and hence the power consumption of the nodes.

We call this variant as site percolation; Haas et al [1] proposed such a gossip protocol and used it for transmitting routing updates in AODV. The terminology is derived from *percolation theory* [20], an old branch of mathematics that studied the residual connectivity of a graph when some fraction of nodes are removed at random from it. Haas, et al., [1] showed that a gossip probability p that is higher than a threshold regime of between 0.6 and 0.8, is sufficient to ensure that almost every node in the network gets the broadcast message with high probability. In other words, if p is lower than this threshold, very few nodes receive the broadcast. However, as p crosses this threshold, there is an abrupt phase transition and all nodes receive the broadcast with high probability. Note that a gossip probability of $p = 1.0$ makes site percolation equivalent to flooding.
Analysis: If p is the gossip probability with which each of the nodes decide independently to transmit a message, then the expected number of messages sent per node in the network is given by, $S_{site} = \frac{1}{N} \cdot \sum_{i=0}^{N} i \binom{N}{i} p^i (1-p)^{N-i} = p$. The average number of messages received per node during broadcast are given by, $R_{site} = \sum_{i=0}^{k} i \binom{k}{i} p^i (1-p)^{k-i} = kp$.

The power consumed by the nodes is thus:

$$\begin{aligned} P_{site} &= \frac{(3 \cdot N \cdot p + kp \cdot N) * Pow}{N} \\ &= (3 + k) \cdot p \cdot Pow \end{aligned}$$

2.4. Bond Percolation-based Gossiping

In percolation theory, the counterpart of site percolation is bond percolation. If k is the number of neighbors for a node in the network, then a node receiving a message will forward the message to only m ($m \leq k$) of these neighbors, picked at random. Like site percolation, bond percolation also shows a phase transition behavior. Specifically, only when the quantity $\frac{m}{k}$ is higher than a threshold value of about 0.5, does the broadcast reach all nodes with high probability [20].

In reality, however, a message transmitted by a node will be received by all its neighbors that are *awake* at that point of time. Thus, a real implementation of this scheme, called PBBF [4] (Probability-Based Broadcast Forwarding) modifies the periodic sleep schedule of each node, so that it stays awake with some probability during the sleep part of the schedule. Such a node can immediately rebroadcast any message it receives (with some probability), and this message will be received only by those neighbors that also happen to be opportunistically awake at that time. We call this variant as modified bond percolation.

For the sake of comparison with the other schemes, we assume that in modified bond percolation, m out of k neighbors receive the broadcast message. The main difference between modified bond percolation and bond percolation is that each node transmits only 1 broadcast message in the former, and m copies in the latter.

Analysis: In bond percolation, because each node sends m messages, we have $S_{bond} = m$. However, for the case of the modified bond percolation, each node will send only one message. Therefore, the number of messages sent per node in this case is $S_{mod-bond} = 1$.

The average number of messages received per node is the same for both above variants of bond percolation, and is given by $\sum_{i=0}^{k} i \binom{k}{i} \left(\frac{m}{k}\right)^i \left(1 - \frac{m}{k}\right)^{k-i} = m$

The power consumed by each of the bond percolation mechanisms can be given as:

$$\begin{aligned} P_{bond} &= \frac{(3mN + mN) \cdot Pow}{N} \\ &= 4m \cdot Pow \\ P_{mod-bond} &= \frac{(3N + mN) \cdot Pow}{N} \\ &= (3 + m) \cdot Pow \end{aligned}$$

2.5. Conclusions from Analysis

The tradeoffs among the four different schemes with respect to energy consumed is illustrated in Figure 1, and explained below. In Figure 1, each arrow is directed from the left-hand side to the right-hand side of an inequality. The expressions next to the arrows indicate the conditions under which the inequality is satisfied.

1. $P_{site} \leq P_{flooding}$, if $p \leq 1$. This is always true as p is a probability.

2. $P_{mod-bond} \leq P_{flooding}$, if $m \leq k$. This is always true, as modified percolation chooses an m that is always lower than k.

3. $P_{mod-bond} \leq P_{bond}$, if $m \geq 1$. This is always true for all practical purposes, since at least one neighbor receives the broadcast message in bond percolation mechanisms.

4. $P_{site} \leq P_{bond}$, if $(3+k).p.Pow \leq 4m.Pow$ or $p \leq \frac{4m}{3+k}$. Choosing $p = 0.7$ (a value between 0.6 and 0.8, as suggested by [1]), and $\frac{m}{k} = 0.5$ (as suggested by [4]), we require $k \geq 2$ for $P_{site} \leq P_{bond}$ to be true. However, $k < 2$ implies a network of disjoint node pairs, which is unlikely in actual deployments. Therefore, $P_{site} \leq P_{bond}$ for all practical networks.

5. $P_{flooding} \leq P_{bond}$, if $(3 + k).Pow \leq 4m.Pow$ or $m \geq \frac{3+k}{4}$. Again, by choosing $p = 0.7$, and $\frac{m}{k} = 0.5$, we have $k \geq 3$ for $P_{flooding} \leq P_{bond}$ to be true. Examples of networks with $k < 3$ can be either a ring or a linear topology. We believe these are unlikely to occur in real deployments. Thus, for all other practical networks we can conclude that $P_{flooding} \leq P_{bond}$.

6. $P_{mod-bond} \leq P_{site}$, if $(3 + m).Pow \leq (3 + k).p.Pow$ or $p \geq \frac{3+m}{3+k}$. For $p = 0.7$, and $\frac{m}{k} = 0.5$, we get $k \geq 4.5$ for $P_{mod-bond} \leq P_{site}$. In other words, site percolation consumes lower energy than modified percolation when the network is sparse, i.e., has typical node degree ≤ 4.

From our discussions, we conclude that modified bond percolation consumes the lowest power per broadcast for dense networks, while site percolation consumes the lowest power for sparse networks. Furthermore, irrespective of the density of the networks, bond percolation consumes the most power, followed by flooding.

3. Simulation Results

While the analytical studies in the previous section looked at the energy consumption, we wish to further study the latency-energy tradeoffs among the schemes. In this section, we compare the four broadcast mechanisms using custom simulations in MATLAB across these different network topologies. As discussed 2.1, we use a MAC independent link layer based on actual traces from [16]. We have replicated the probability of successful packet reception for the medium and the low power scenarios from [16] in Figure 2. The plot shows that the probability of successful packet reception as a function of distance and transmit power. While the term 'power' actually implies the transmit power in [16] (that is controlled using a potentiometer), the power

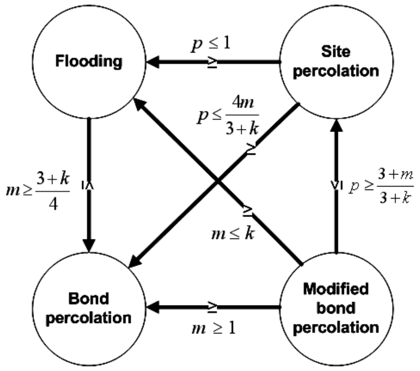

Figure 1. **Conclusions on energy consumed by the broadcast mechanisms (arrows are directed from the LHS to the RHS of the embedded inequality, which is true when the condition next to the arrow is satisfied).**

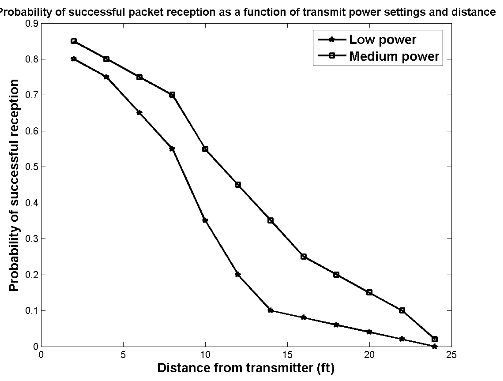

Figure 2. **Probability of successful packet reception as a function of distance and power.**

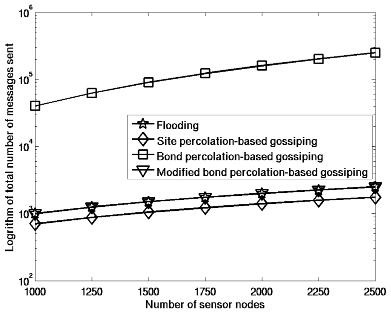

Figure 3. **Number of messages sent in the network - random topology.**

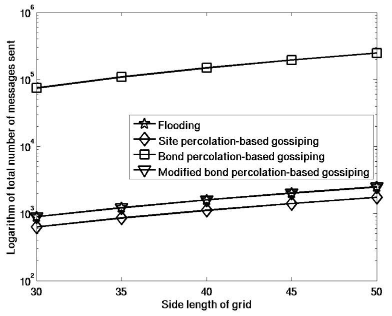

Figure 4. **Number of messages sent in the network - grid topology.**

consumption values used for comparison in this paper (both in this section and in Section2) is written in terms of the receive power, Pow. We can therefore consider Pow to be the receive power, when the transmit power is set as in [16].

Experimental Methodology: We simulate three different topologies, namely: (1) random, (2) clustered, and (3) grid. For the random topology, we distribute the nodes uniformly at random on a 50×50 area. For the clustered topology, we create four square clusters, each with a side 20 on the four corners of a 50×50 square. Nodes are distributed uniformly at random within each cluster, and are shared evenly by the four clusters. The number of nodes, for both the random and the clustered topology, is varied from 1000 to 2500 in steps of 250. In the case of the grid topology, we vary the grid size from 30 to 50 in steps of 5. The number of nodes in the grid topology is the square of a side of the grid, and hence varies from 900 to 2500 (which is comparable to that for the random and the clustered topology). While the random and clustered topologies are unplanned networks, a grid topology is a planned network. We performed our simulations for both the medium power and low power scenarios (corresponding to a potentiometer setting of 66 and 69, respectively) discussed in [16]. We have, however, provided the results only for the low power scenario in this paper. The results for the medium power characteristics follow a similar trend as that of the low power case.

The minimum node degree (average number of neighbors) for the low power model turned out to be 15 for random topology, 10 for the clustered topology, (corresponding to 1000 nodes) and 30 for the grid topology (corresponding to grid size 30). Since these are dense networks, we also simulate a sparse grid topology with a smaller node degree (around 4 for a grid size of 30) by choosing a transmission range of 1. For site percolation, we choose a gossip probability p of 0.7, as it lies between the threshold regime of 0.6 and 0.8, suggested in [1]. For the two bond percolation schemes, we choose the $\frac{m}{k}$ ratio (where m is the number of nodes that receive the broadcast message out of k neighbors) to be the exact threshold value of 0.5, as mentioned in [4]. We generate 100 different realizations of the network for each of the three topologies, each with a different seed. During each realization, we pick a node uniformly at random, and

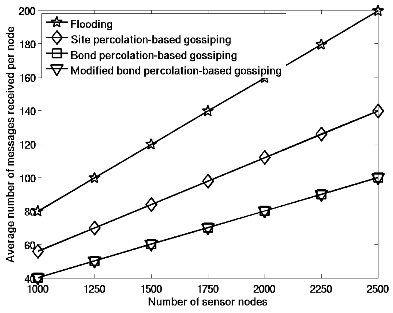

Figure 5. **Average number of messages received per node - random topology.**

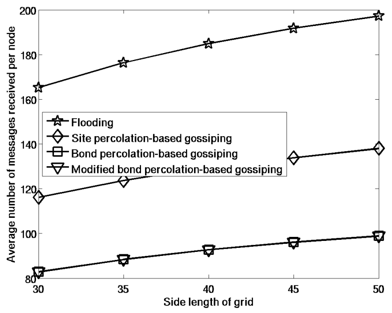

Figure 6. **Average number of messages received per node - grid topology.**

generate a broadcast message from this node for propagation throughout the network. Furthermore, we generate different link characteristics for each realization.

We simulate all the four broadcast mechanisms explained in Section 2. Our metrics of interest are: number of messages sent, number of messages received, average energy consumed, latency of receiving broadcasts (since broadcast initiation time). We initiate broadcasts from only one sender node. Our baseline for the comparison is to set parameters for the four broadcast schemes, so that they just barely ensure 100% reliability for all broadcast messages. In this paper, due to space constraints we provide the results only for the random and grid topologies. Interested readers may refer the results for the clustered topology from [21].

Message and Bandwidth Consumption: Figures 3 and 4 show the plots for the logarithm of the number of messages sent in the network during the broadcast process in the random and the grid topologies respectively. We can see from the plots that for bond percolation, the number of messages sent grows very quickly as the number of nodes increase. This is because each node has to send m individual messages. However, the other three percolation schemes are almost identical, except that site percolation sends slightly fewer messages.

The average number of messages received per node is plotted for the random and grid topologies in Figures 5 and 6. We observe that the average number of messages received per node is the lowest for the two bond percolation schemes. We can also note that the average number of messages received for the site percolation is lower than that of flooding, due to the probabilistic broadcast nature of the site percolation scheme.

Latency: Next, we plot in Figures 7 and 8, the latency involved in broadcast for the two topologies. The time values are normalized to the units of T, where T is the time taken to complete one transmission from one node to its neighbor, and therefore are not absolute latency values. We observe that flooding is the fastest, while site percolation is only slightly slower. However, the latency for the two bond percolation schemes is high since fewer nodes receive the message per broadcast. This leads to longer propagation times for broadcasting the message to the entire network.

Energy Consumption and Tradeoffs: Similar to our analysis (see Section 2.1), we measure the power consumed by a broadcast packet, which is the energy consumed by a normalized, one second long packet. Figures 9 and 10 show the units of average receive energy (or power) and the total (sum of receive and transmit) energy consumed by the nodes during the broadcast process (in units of $Pow = P$, see Section 2.1). As shown by the figure, the bond percolation scheme consumes more transmit power, as it is required to make m individual transmissions. This is evident from the plot for the total power consumed. However, the received power is very low for both the bond percolation schemes. As a result, the modified bond percolation scheme consumes the least total power among all the broadcast schemes. Site percolation consumes lower power than flooding.

To show that site percolation may consume lower energy than a modified percolation when the number of neighbors, $k \leq 4.5$, we simulate a sparse grid topology with the same number of nodes as before (900 to 2500). However, we reduce the transmission range for each node to just 1. This ensured that each node has only 4 neighbors on average. The corresponding plot is shown in Figure 11. We observe from the figure that the total power consumed by site percolation is slightly lower than modified bond percolation, as predicted by our analysis.

Finally, in order to map the tradeoffs among the four broadcast mechanisms, we vary the number of nodes in the system, and plot the latency as well as total energy consumed per-node during the entire broadcast propagation. This is shown in Figures 12 and 13. In these figures, each point corresponds to a varying number of nodes from left-to-right and bottom-to-top (1000 to 2500 for random topology, and 30 to 50 for grid topology, as described earlier). We can see that the modified bond percolation scheme consumes the lowest power, but incurs a high latency. In comparison, the flooding and site percolation schemes consume more power

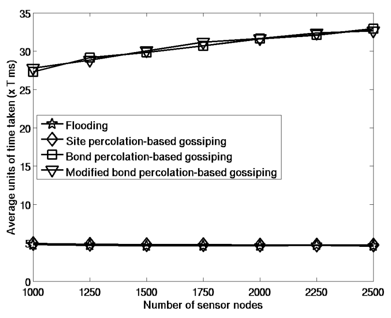

Figure 7. **Average units of time taken - random topology.**

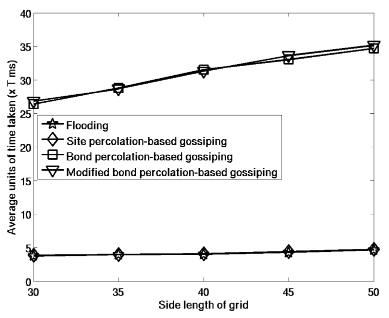

Figure 8. **Average units of time taken - grid topology.**

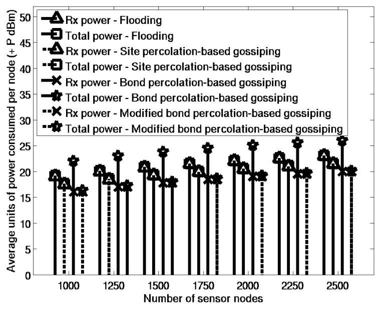

Figure 9. **Average units of power consumed - random topology.**

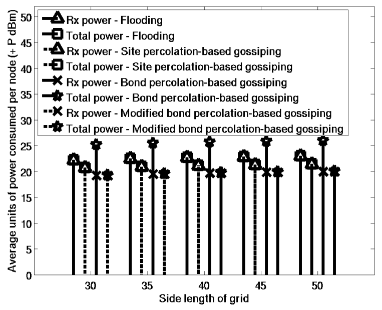

Figure 10. **Average units of power consumed - grid topology.**

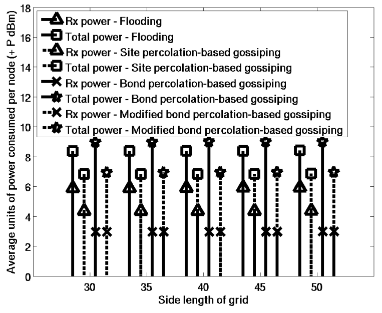

Figure 11. **Average units of power consumed - grid topology, with Tx range = 1.**

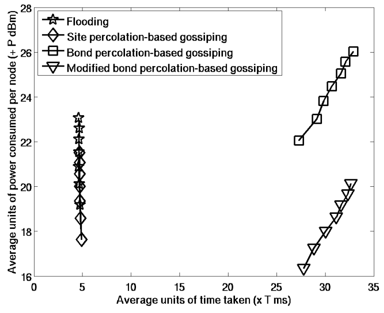

Figure 12. **Latency vs. power consumed - random topology.**

but propagate broadcasts faster. Further, bond percolation consumes more energy and takes much longer to propagate the broadcast to all nodes. Figure 14 shows the energy-latency tradeoff for a grid topology with transmission range of 1. We observe that site percolation consumes lower power and is faster than modified bond percolation mechanism. Flooding and bond percolation perform similar to that in Figure 13.

From the above discussion, we can conclude that the modified bond percolation (for dense networks) and site percolation (for sparse networks) are useful for applications that are required to be energy-efficient, while site percolation and flooding are useful for applications that require low latency.

4. Implications and Caveats of our Study for Practical Broadcast Operations

4.1. Implications

We now discuss three different applications of the broadcast mechanisms in sensor networks, and the implications of our study for these applications. Our findings are summarized in Table 1.

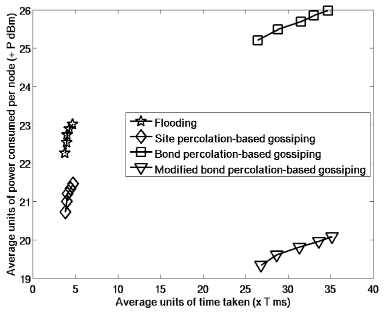

Figure 13. **Latency vs. power consumed - grid topology.**

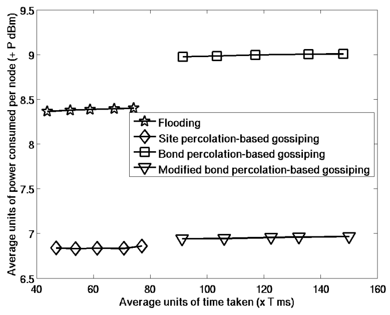

Figure 14. **Latency vs. power consumed - grid topology, with Tx range = 1.**

Code updates: The sensor nodes in a network may require frequent code updates for changing the tasks that these nodes have to execute [3]. During the code update process, each node with a stale code has to update itself and propagate the update to its neighbors. Such a code update process require high degree of reliability, as the sensing task may fail otherwise. However, a code update process is not time sensitive, although reducing the broadcast propagation latency and the total energy consumption always helps.

From our analysis, we can see that by choosing suitable parameters, high reliability can be obtained using any of the four broadcast schemes. However, to reduce the energy consumption, percolation-based protocols may be preferable compared to flooding for dense networks. Flooding may be good if the network is sparse and clustered, or if the sensor nodes are not energy-constrained, or if the code updates are infrequent.

Query and information propagation: Sensor networks need to be queried about their current status, for example, when average temperature is required to be calculated across a region for a histogram of temperature readings [5]. Such instantaneous querying requires timely propagation of query messages to the sensor nodes. Furthermore, the information sent by the sensor nodes in response to the query messages has to be routed back to the query source with soft real-time requirements. This means that, irrespective of the density of the network, flooding or site percolation-based gossiping is preferable compared to other approaches, for query propagation. Flooding may be more preferable if query correctness is highly sensitive to propagation time, and energy is aplenty.

Membership management: Since sensor networks need to be self-sufficient, groups of sensor nodes may need to run a membership management protocol. For instance, this group may be comprised of all nodes that have a particular type of sensor (e.g., acoustic sensor, for tank detection), or sensors that are in a specific geographical region. Each node in the group maintains a (partial) list of other nodes from the group that are currently awake. This information can then be used for multicast or aggregation within the group. The bootstrapping and maintenance of the membership list at each node can be maintained via broadcasts from each node as it goes to sleep and wakes up. The requirement for continuous and timely broadcasting means that energy consumption needs to be minimized - this points to modified bond percolation (for dense networks) and site percolation (for sparse networks) being the best choices for membership management.

4.2. Caveats

Effect of MAC collisions: Since we ignore MAC layer collisions and contentions in our simulations and analysis, the results are in a sense positively biased to flooding when compared to the different percolation approaches. This is because flooding generates many more messages, both overall and collision-wise. However, because our final results show that percolation approaches are preferable to flooding, we believe the same conclusions will hold even if specific MAC protocols are taken into account.

Effect of on-off duty cycle: Our simulation results assume a common on-off duty cycle for all the sensor nodes, and across all the broadcast approaches compared. However, in reality, the duty cycles may vary across all the sensor nodes. Furthermore, depending on the duty cycles different broadcast mechanisms may be preferred. For instance, bond percolation is suitable for low duty cycle sensor networks in which few neighbors wake-up simultaneously. The sender can, therefore send the broadcast message to a subset of neighbors. On the other hand, site percolation will be suitable for sensor networks with faster duty cycles, as a single broadcast is intended for all the neighbors of the sender. The results compared above may thus be different depending on the duty cycles. A common duty cycle for the all the mechanisms is, however, chosen to provide a common base for comparison.

Table 1. **Appropriate broadcast mechanism for each of the applications - further choices depend on tradeoffs involved in the application, and the network layout.**

Applications	**Requirements**	**Flooding**	**Site percolation**	**Bond percolation**	**Modified bond percolation**
Code updates	High reliability		√	√	√
Query propagation	Low latency	√	√		
Membership management	Low power		√		√

5. Conclusion

In this paper, we have analyzed four fundamental percolation-based broadcast mechanisms in sensor networks, namely: flooding, site percolation-based gossiping, bond percolation-based gossiping, and modified bond percolation algorithms. Our analytical and simulation results show that flooding and site percolation schemes have low latency. For dense networks, modified bond percolation consumes the lowest energy per broadcast, while site percolation consumes the lowest energy for sparse networks. Furthermore, our analysis also shows that flooding involves a larger number of messages during the broadcast propagation. These conclusions hold for three types of topologies that practical deployments use - random, clustered, and grid. We also identified the tradeoffs of these broadcast alternatives for code updates, query propagation, and membership management.

References

[1] Z. Haas, J. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE Infocom*, 2002.

[2] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," *ACM Mobicom*, 1999.

[3] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks," *USENIX NSDI*, 2004.

[4] M. Miller, C. Sengul, and I. Gupta, "Exploring the energy-latency tradeoff for broadcasts in energy-saving sensor networks," *ICDCS*, 2005.

[5] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *OSDI 02*, December 2002.

[6] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *MobiCom*, 2000.

[7] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *ICENSS*, 2004.

[8] A. Ahmed, H. Shi, and Y. Shang, "A survey on network protocols for wireless sensor networks," *ITRE.*

[9] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Communications*, December 2004.

[10] Q. Jiang and D. Manivannan, "Routing protocols for sensor networks," *CCNC*, January 2004.

[11] M. Akdere, C. Bilgin, O. Gerdaneri, I. Korpeoglu, O. Ulusoy, and U. Cetintemel, "A comparison of epidemic algorithms in wireless sensor networks," *Computer Communications*, vol. 29, no. 13, 2006.

[12] L. Luo, T. F. Abdelzaher, T. He, and J. A. Stankovic, "Design and Comparison of Lightweight Group Management Strategies in EnviroSuite," in *DCOSS*, 2005.

[13] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," *IEEE Infocom*, April 2003.

[14] S. Ray, R. Ungrangsi, F. D. Pellegrini, A. Trachtenberg, and D. Starobinski, "Robust location detection in emergency sensor networks," *IEEE Infocom*, 2003.

[15] S. Meguerdichian, S. Slijepcevic, V. Karayan, and M. Potkonjak, "Localized algorithms in wireless adhoc networks: Location discovery and sensor exposure," *ACM Mobihoc*, October 2001.

[16] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, "Complex behavior at scale: An experimental study of low-power wireless sensor networks," *UCLA CS Technical Report*, 2002.

[17] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," *SenSys*, 2004.

[18] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," *SenSys*, 2003.

[19] J. Polastre, R. Szewczyk, C. Sharp, and D. Culler, "The mote revolution: Low power wireless sensor network devices," *http://webs.cs.berkeley.edu/papers/hotchips-2004-motes.ppt.*

[20] G. Grimmett, "Percolation," *Springer-Verilag*, 1989.

[21] V. Raman and I. Gupta, "Performance tradeoffs among percolation-based broadcast protocols in wireless sensor networks," in *Dept. of CS Technical Report, UIUC*, March 2009.