Idle-listening Reduction for Data Aggregation in Distributed Sensor Networks

Hongchao Zhou, and Xiaohong Guan, *Fellow, IEEE*

Abstract—For existing sensor systems, the power consumption of radio for *idle listening* is in the same order with that for transmitting. In order to prolong the lifetime of sensor networks, we propose a reliable and energy-efficient data aggregation scheme (Reed) to reduce *idle listening* time. In this application scenario, each sensor node periodically detects the environment and is allowed to send one and only one data packet in each sampling period T. Based on this property, the *idle listening* time can be significantly reduced by node scheduling, while the network routing topology is maintained. Three techniques are used in Reed: First, the sensor nodes in the network are divided into two types: dominating nodes and non-dominating nodes. Non-dominating nodes sleep for most of the time and only need to awake to broadcast sensing data periodically. Dominating nodes maintain network topology and aggregate sensing data to the sink. Second, role rotation is used to balance the energy consumption of dominating nodes and non-dominating nodes. Third, for the dominating nodes, the *idle listening* time can be reduced further by predicting the incoming time of packets from neighbors. Analysis and simulation results show that Reed can achieve high reliability and can significantly reduce the energy consumption. It was shown that when the sampling period $T = 10s$, the energy consumption of Reed is about 70 times less than the network in which all nodes keep active.

Index Terms—distributed sensor networks, reliable and energy-efficient data aggregation, duty cycle.

1 Introduction

Wireless sensor networks consisting of a large number of sensor nodes are widely used in area monitoring, such as temperature and humidity monitoring in a specific area. Each sensor node comprises one or several sensors, one processor, and one radio. In this paper, we consider data aggregation scenario, where each sensor node periodically generates sensing data and transfers these data to the sink via multiple hops.

Energy is the most crucial resource in sensor networks, especially for the application where the sensor nodes have to work for a long period without replacing batteries. As shown in [1], we know that most of energy in sensor networks is consumed by radio, and the power consumption for *idle listening* is in the same order for transmitting. For example in Mica2 (sensor node developed by UC Berkeley) the current consumption for transmission is from 3.7mA to 21.5mA, depending on the transmission power. But the current consumption for listening is 7mA regardless of transmission activity. If the radio keeps listening for incoming messages, it will cost most of the battery energy.

Considering the feature above, in many existing systems the sensor node ″sleeps″ for most of the time and only wakes up when it needs to transmit its own packet. For example, a sensor system was developed by UC Berkeley to measure the traffic on a freeway or at a street intersection [2]. In this system, the power consumption is 1,000 times lower than a content network, in which each node keeps listening for incoming messages. In [3], a sensor network for habitat monitoring was deployed on a remote island off the coast of Maine for four months. In both of these systems, the lifetime of networks was significantly prolonged by reducing the idle listening time of radio. However, they only consider a network with very simple topology that the access point (sink) can send messages to all of sensor nodes in one hop and their methods cannot directly apply to the distributed sensor networks.

In this paper, we propose a reliable and energy-efficient data aggregation protocol Reed for distributed multi-hop sensor networks, where the communication range of the sink and sensor nodes is limited and data is transmitted via multiple hops. In Reed, we divide all the nodes into two types: dominating nodes and non-dominating nodes. Non-dominating nodes can ″sleep″ for most of the time and only need to wake up when they report their sensing data. Dominating nodes form a routing tree rooted at the sink such that all the sensing data can be aggregated along the tree [4]. Since the energy consumption of dominating nodes is much more than that of non-dominating nodes, we hope that the number of dominating nodes can be as a few as possible, while all the non-dominating nodes can be covered by the dominating nodes (each non-dominating node should have at least one dominating neighbor). In order

- *Hongchao Zhou is with the Distributed Information Systems Group, Electrical Engineering Department, California Institute of Technology, Pasadena, USA. Most of this work was done when he was with the Center for Intelligent and Networked Systems, Tsinghua University, Beijing, China.*
  E-mail: hzhou@caltech.edu
- *Xiaohong Guan is with the Center for Intelligent and Networked Systems, Tsinghua University, Beijing, China. He is also with the Systems Engineering Institute, Xian Jiaotong University, Xian, China.*

Manuscript received July 30, 2009.

to prolong the lifetime of the network further, we need: (1) Balancing the energy consumption of dominating nodes and non-dominating nodes. This can be achieved by rotate the roles of each sensor node. (2) Reducing the power consumption of dominating nodes. Since each sensor node is allowed to send one and only one packet in each sampling period in our application scenario, it is possible for dominating nodes to predict the arriving time of each incoming message. Based on this prediction, they can make a *sleep-listening* schedule to reduce the idle listening time as well as power consumption.

Our method is different from a kind of methods that called area coverage, as shown in literature [22]–[25]. In area coverage, in order to save energy, sensor nodes are turned off as many as possible. The nodes turned off do not detect the environment periodically, as a result, a large amount of sensing data is lost and the detection accuracy is reduced. In reed, we only want to reduce the useless idle listening time to prolong the network lifetime without reducing the detection accuracy. Another related works are duty-cycle MAC protocols [30], [31]. Duty-cycle MAC protocols are more energy efficient than traditional MAC protocols. However, they have to synchronize different nodes accurately or introduce a long preamble before transmission, which may increase the traffic of the network. In the following sections, we can see that Reed can work under the simplest CSMA/CA MAC protocol and can significantly reduce the energy consumption in the networks.

The remainder of this paper is organized as following: In section 2, we introduce existing related works. In section 3, we present Reed and discuss the implementation of Reed in the real world. Section 4 analyzes the power consumption of each node in Reed and compare it with the power consumption of other protocols. The reliability of Reed under unreliable wireless links is simulated and evaluated in section 5 followed by the conclusion.

2 Related Works

In wireless sensor networks, several topology management protocols have been proposed to turn off redundant sensor nodes without disregarding the network connectivity. In Geographic Adaptive Fidelity (GAF) [5], the nodes are divided into small groups based on virtual grids. At each time, only one node is required to be active within each group and the others in the group keep sleeping. Each virtual grid is a square with r units on a side, in order to maintain the network connectivity r and the communication range R have the following relation: $r \leq \frac{R}{\sqrt{5}}$. The limitation of GAF is that each node should know its location, by GPS or other location systems. However, in most of area monitoring application, sensor nodes do not have their position information, or their position information is not accurate enough. In AFECA [6], a constant density of active nodes are maintained by periodically turning radio off, where the sleeping time is proportional to the number of neighbors. In the network, the number of active nodes is roughly constant, so as the density increases, more nodes will be turned off and more energy will be saved. The work in [7] and [8] provides us the asymptotic result on the relationship between the communication range and the network connectivity. Using percolation theory, it proves that for a random network with size n, in order to maintain network connectivity, the average node degree of the network should be in the order of $(\log n + c)$, where c is a constant. Our method is similar with SPAN [9], which is a distributed, randomized algorithm. In SPAN, a backbone is maintained and the nodes in backbone are called coordinators. SPAN lets the nodes that do not belong to the backbone to sleep. Since the energy cost of a coordinator is much more than that of a non-coordinator node, periodically a non-coordinator node determines if it should become a coordinator or not. By this way, the energy consumption among different nodes is balanced. However, SPAN is designed for multi-hop ad hoc wireless networks to reduce energy consumption without significantly diminishing the capacity or connectivity of the network. As what we will see in the following sections, SPAN cannot be applied to our data aggregation scenario directly.

Another kind of related works are to compute the smallest connected dominating set (CDS), where each node in the network either belongs to CDS or has at least one neighbor in CDS. Computing CDS is known a NP-hard problem and requires global information of the network topology [10]. In order to reduce the number of relaying nodes in a broadcasting task, distributed algorithms to get efficient CDS have been proposed [4]. Wu and Li [11] proposed a marking scheme to construct CDS in ad hoc networks, in which two pruning rules are used to reduce the size of CDS. Dai and Wu [12] extent the two rules into a more general pruning *Rule k*, in which a dominating node becomes a non-dominating node if all of its neighbors are covered by an arbitrary number of connected 1-hop dominating neighbors. Later, many algorithms based on Rule k are developed. In order to avoid simultaneous withdrawals in mutual coverage cases, Stojmenovic [13] studied reducing the size of CDS via adaptive interpretation of priority values. Wu et al. [14] used an iterative local solution to compute a CDS, which applied Rule k to reduce the size of CDS round by round.

However, none of the above algorithms have considered the problems of reducing idle listening time of radio. Ma et al. [15] have studied this problem in a two-layer sensor networks. In their solution, a number of powerful nodes called cluster heads are deployed in the field. Each cluster head works as a local sink, and organizes nearby nodes into the cluster. Comparing with their work, our work explores this problem in multi-hop sensor networks, rather than two-layer centralized sensor networks. Some other methods are also proposed to reduce the energy consumption in data aggregation for area monitoring. For example, according to the cor-

TABLE 1
Current Reqirements for Different Sensor Nodes under Various Operations

Operation Current(mA)	IRIS [16]	MICAz [16]	MICA2 [16]	BTNODE [17]	TmoteSky [18]
Processor on, Radio off	8	12	12	12	1.8
Processor Idle, Radio off	0.008	0.010	0.010	3	0.055
Radio, receive	16	19.7	7	32	21.8
Radio, transmit	17(1 mW power)	17(1 mW power)	10(1 mW power)	32	19.5

relation of data, different coding methods can be used to compress the amount of reported data [19] [20] [21]. These methods are independent with Reed and can work with Reed together to prolong the network lifetime.

3 Energy-preserving Scheme

By now, several literatures have analyzed the hardware characteristics for low-cost sensor nodes, such as [1] and [15]. Table 1 presents the energy consumption characteristics under various operation modes on different hardware platforms: IRIS, MICAz, MICA2 [16], BTNODE [17], TmoteSky [18]. We find that (1) The energy consumption of radio in receiving messages is very close to that in idle-listening, since in these two modes, most of the energy is consumed by the electrical circuit modules, regardless of the RF signal strength. (2) The energy consumption for radio to transmit messages is close, at least in the same order, to the energy consumption for radio to receive messages. Sometime the latter one can be greater in some systems. Existing sensor nodes are often powered by AA alkaline battery that can provide 1.5 volts, $1800 \sim 2600$ mAh electrical output. One sensor can at most survive for about 100 hours if keeping its radio in listening/receiving/trasmitting modes [15]. Based on the facts above, we can find that reducing idle listening time of radio is necessary.

In this section, we firstly present Reed under ideal links, where we assume that all the connections are reliable and bidirectional: if node u is a neighbor of node v then node u and v can communicate with each other with high probability, otherwise u and v cannot communicate with each other. Reed is based on three ideas: (1) There are many nodes (non-dominating nodes) in the network that do not need to listen for incoming messages, therefore they can "sleep" for most of the time. (2) For other nodes (dominating nodes) , they can predict the arriving time of incoming messages in the next period and reduce the idle-listening time based on this prediction. (3) Role rotation can be used to balance the energy consumption of different nodes and prolong the network lifetime. Finally, we discuss that how to implement Reed in the real world, where the links are lossy and asymmetric.

3.1 Data Aggregation Based on Dominating Tree

In Reed, each node detects the environment periodically with sampling period T and all of the sensing data needs

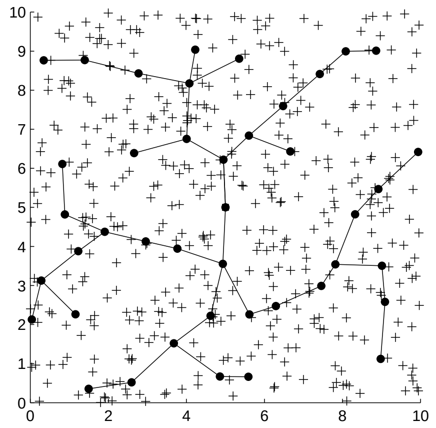

Fig. 1. An illustration of dominating nodes and non-dominating nodes. '•' is for dominating nodes, and '$+$'is for non-dominating nodes.

to be aggregated to the sink. In order to save energy, we divide the nodes into two types: dominating nodes and non-dominating nodes, as shown in Fig.1. Non-dominating nodes do not need to listen for incoming messages, but each of them should be able to send messages to at least one dominating neighbor. Each node has a local clock. After each period, it detects the environment and reports the sensing data to its neighbors, then it "sleeps" for the rest of the period. The dominating nodes form a tree rooted at the sink such that it covers the whole network, called dominating tree. After each period, it fuses the data from its dominating successors and non-dominating neighbors, then it detects the environment and reports the fused data to its parent. In order to save energy, non-dominating nodes do not know neighbor information, it may broadcast the same sensing data to different dominating neighbors. In this case, duplicated sensing data should be eliminated during data aggregation and fusion.

At the beginning, we assume that all of the nodes are in dominating state and each node knows its "distance" to the sink, where this abstract distance can be Euclidean distance, minimal number of hops, or other definitions. Using vector-based algorithms, such as Geographic Forwarding and Gradient-based Routing, each node selects its neighbor "closest" to the sink as parent to forward messages, therefore a dominating tree is constructed. Now, we turn some dominating nodes into non-dominating state and hope the number of dominating nodes can be as a few as possible. The problem is how to determine that whether a dominating node can be turned into non-dominating state, while the other dominating nodes still form a dominating tree. We notice that for a dominating node u, it cannot change its state to non-dominating state if it satisfies either of the following conditions:

1) There exists a dominating node v, such that v has only one dominating neighbor closer to the sink, that is u.
2) There exists a non-dominating node v, such that v has only one dominating neighbor, that is u.

Condition 1 can ensure that the rest of the dominating nodes can connect to the sink after removing a dominating node. Now, we give a stronger conclusion: let's use digraph $G = (V, E)$ to describe the network constructed by the dominating nodes, where $(u, v) \in E$ iff both u and v are dominating nodes and v is a neighbor of u closer to the sink, then we have

Theorem 1. *Given a loop-free digraph $G(V, E)$ in which all of the nodes can connect to the sink. For a node $u \in G$, if $\forall$ node v with $(v, u) \in E$, there exists a node $w \neq u$ such that $(v, w) \in E$, then after removing node u, the resulting digraph is still a loop-free digraph in which all of the nodes can connect to the sink.*

Proof: Let $G'(V', E')$ denote the digraph after removing node u, where $V' = V/u$. It is obvious that there are no loops in G', and we only prove that all of the nodes in G' can connect to the sink.

In $G'(V', E')$, we use $A \to B$ to denote that there exists a path between A and B; otherwise, $A \nrightarrow B$. Let's prove the conclusion in the theorem by contradiction. Assume there exists one node which cannot connect to the sink in G' and $S = \{v | v \in G, (v, u) \in E, v \nrightarrow sink \text{ in } G'\}$.

If $S = \phi$, then for each node we can always find a path to the sink in G'.

If $S \neq \phi$, then we will show that there must be some loops among the nodes in S, which contradicts with our assumption. For each node $v \in S$, there exists a node $w \neq u$ such that $(v, w) \in E$. We know that $w \nrightarrow sink$, otherwise we can find a path to connect v and the sink. Therefore, there exists a node $l \in S$ such that $w \to l$. Then we have that for each node $v \in S$, there exists a node $l \in S$ such that $v \to l$, where l and v can be the same node or not.

Now we show that there must be some loops among the nodes in S. Now, let each node in S choose another node in S to connect to. In order to avoid loops, the $1st$ node has $|S|-1$ choices (it cannot connect to itself, otherwise there will be a loop); choosing the node connected by the $1st$ node as the $2nd$ node, it has $|S| - 2$ choices ... finally, we find that the last node has no choices, where contradiction happens. So we can conclude that all of the nodes in G' can connect to the sink. □

Condition 2 can ensure that all of the non-dominating nodes can report their sensing data to at least one dominating neighbor. The following theorem tells us that it is difficult for a dominating node to determine whether all of its non-dominating neighbors can be covered by other dominating nodes, but it is much easier to determine whether its non-dominating neighbors can be covered by its dominating neighbors.

Theorem 2. *Assume non-dominating nodes do not have information about neighbors. There does not exist a constant integer K such that for all networks in which each dominating node can determine whether all its non-dominating neighbors can be covered by other dominating nodes, based on the neighbor information in K-hops.*

Proof:

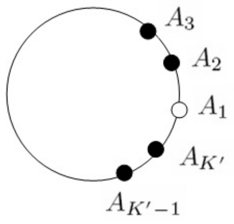

Fig. 2. An illustration for the proof of theorem 2.

Proof: Assume that there exists such integer K. We can construct a network with $K' > K+2$ nodes as shown in Fig.2, where each two neighbors can communicate with each other. Node A_1 is a non-dominating node and the other $K' - 1$ nodes are dominating nodes. In this graph, if A_2 wants to know whether it satisfies Condition 2 or not, it must know the node $A_{K'}$'s information. Since there are $K' - 2$ hops between node A_2 and $A_{K'}$, node A_2 needs information beyond K hops, which contradicts with our assumption. □

3.2 Role Rotation

Using the method in the subsection above, all the nodes in the network can be divided into dominating nodes and non-dominating nodes. Since the energy consumption of dominating nodes is much more than that of non-dominating nodes, role rotation between them is necessary to balance the energy consumption among different nodes and further prolong the lifetime of the network. As shown in Fig. 3, one node stays in dominating state for time T_a, where T_a is a random variable, then it decides whether to change state. If it satisfies some rules, it changes its state to non-dominating state and stays in non-dominating state for time T_c. Otherwise, it stays in dominating state for time T_b and re-check whether it satisfied those rules or not. In order to avoid multiple neighbor-nodes changing states simultaneously, announcing state is introduced as a transition state between dominating state and non-dominating state. Announcing state can last for one or two sampling periods. Based on the discussion in the section above and the consideration of announcing state, a dominating node u cannot change

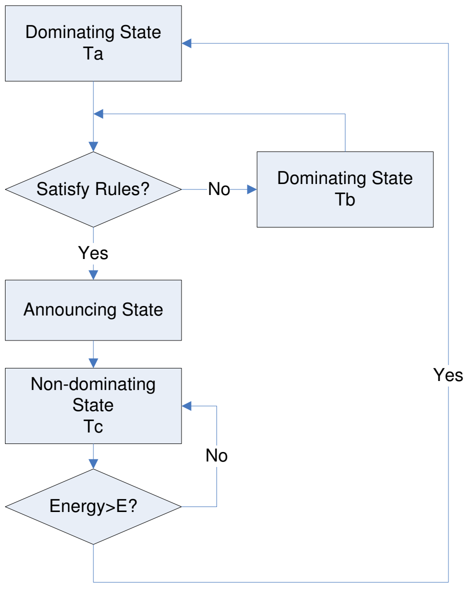

Fig. 3. Schematic diagram of Role Rotation Scheme.

its state from dominating state to non-dominating state if and only if it satisfies at least one of the following conditions.

1) There exists a dominating node v, such that v has only one dominating neighbor closer to the sink, that is u.
2) There exists a non-dominating node v which is a neighbor of u, such that v cannot covered by the dominating neighbors of u.
3) In the last period, node u heard a message from a neighbor in announcing state.

Note that all the conditions above can be checked based on local one-hop information, since each dominating node knows its neighbor information. Therefore, a dynamic dominating tree can be maintained based on local decisions. And more, we notice that different nodes may have unequal energy left in their batteries. If one node's energy left is lower than the threshold energy, this node should always stay in non-dominating state. In this way, this node can work as long as possible (See Fig.3).

In our scheme, both T_a and T_c are random variables. We can see that if $\frac{T_a}{T_c}$ is too large, most of the nodes will stay in dominating state, that will increase the energy cost. If $\frac{T_a}{T_c}$ is too small, there will be some nodes with enough energy staying in dominating state for very short time. Here, let $N(u)$ denote the number of neighbors of node u, if $\frac{T_a}{T_c} \gg 1$ then the expected number of the dominating neighbors of node u is $\frac{N(u)T_a}{T_a+T_c}$. Some results on statistical connectivity of wireless networks suggest that $\frac{N(u)T_a}{T_a+T_c}$ should be slightly smaller than 4, otherwise there will be some redundant dominating nodes. Thus, we can choose T_a, T_b, T_c as

$$T_a = T_b = (1+R)kT$$

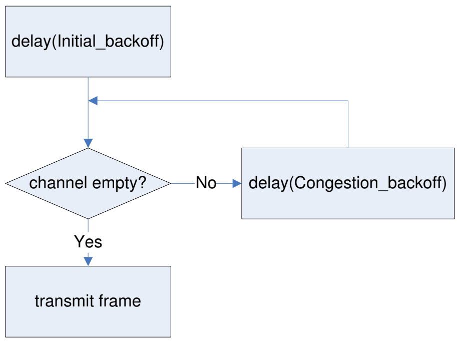

Fig. 4. Schematic diagram of CSMA/CA protocol.

$$T_c(u) = (1+3N(u)R)kT$$

where k is a constant integer and T is the sampling period. The randomization is achieved by picking R uniformly at random from the interval $[0,1]$.

3.3 Energy-saving for Dominating Nodes

So far, the energy consumption of dominating nodes dominates the total energy consumption. In this section, we study that how to reduce the energy consumption of dominating nodes by reducing their *idle listening* time.

We assume that CSMA/CA protocol is used as the MAC protocol in our system, which is widely used in wireless sensor networks to avoid collisions and does not need accurate synchronization among nodes such as TDMA. This protocol repeats to generate a random delay until the channel is idle, then the frame is transmitted. Fig. 4 is a simple schematic diagram of CSMA/CA. *Initial_backoff* is the random time delay introduced at the beginning, to avoid different nodes sending messages at the same time. *Congestion_backoff* is the random time delay introduced if there are some neighbors sending messages simultaneously.

Each node has a local clock, with independent offsets. In the following description, all the time is based on local measurement. Assume one node is asked to send one packet (kth packet) at time $T^{(k)}_{request}$, then the next time that the node is asked to send a packet is

$$T^{(k+1)}_{request} = T^{(k)}_{request} + T$$

Due to MAC layer delay in CSMA/CA, the actual time to send a packet out is often slightly behind the time of receiving request from upper layer, that is

$$T^{(k)}_{send} = T^{(k)}_{request} + T^{(k)}_{I} + \sum_{j=1}^{N(k)} T^{(k)}_{C}(j)$$

where $T^{(k)}_{I} + \sum_{j=0}^{N(k)} T^{(k)}_{C}(j)$ is the random time delay introduced by CSMA/CA. T_I is corresponding to the initial delay, N is the number of congestion delays generated to send the current packet, and $T_C(j)$ is the jth congestion delay. Note that if there are multiple nodes requested to send messages at the same time, then after time T they are still requested to send messages at the same time. In order to avoid congestion in the next period, we can modify $T_{request}^{(k+1)}$ a little if $N(k) \neq 0$, that is

$$T_{request}^{(k+1)} = T_{request}^{(k)} + T + \sum_{j=1}^{N(k)} T_C^{(k)}(j)$$

where $\sum_{j=1}^{N(k)} T_C^{(k)}(j)$ is the time delay introduced by congestion in the last period.

We can see that each node sends messages almost periodically. Assume node A begins to receive the kth packet sent from node B at time $T_{receive}^{(k)}$, which is based on the clock of node A. In order to reduce idle listening time, node A tries to predict the incoming time of the next packet from node B, that is $T_{receive}^{(k+1)}$. Let $T_{send}^{(k)}$ denote the time that node B begins to send the kth packet to node A, then we have that

$$\begin{aligned}
& T_{receive}^{(k+1)} \\
= \;& T_{receive}^{(k)} + T_{send}^{(k+1)} - T_{send}^{(k)} \\
= \;& T_{receive}^{(k)} + [T_{request}^{(k+1)} + T_I^{(k+1)} + \sum_{j=1}^{N(k+1)} T_C^{(k+1)}(j)] \\
& -[T_{request}^{(k)} + T_I^{(k)} + \sum_{j=1}^{N(k)} T_C^{(k)}(j)] \\
= \;& T_{receive}^{(k)} + T + T_I^{(k+1)} + \sum_{j=1}^{N(k+1)} T_C^{(k+1)}(j) - T_I^{(k)}
\end{aligned}$$

In order to predict $T_{receive}^{(k+1)}$, node A should know $T_I^{(k)}$ and predict $T_I^{(k+1)}$ and $T_C^{(k+1)}(j)$. This can be done because $T_I^{(k)}$, $T_I^{(k+1)}$ and $T_C^{(k+1)}(j)$ generated by computer are not real random variables, instead, they are pseudo-random variables generated by pseudo-random generator. So their values depend on the random seed. If the node A knows $N(k)$ and the seed $g^{(k)}$ of $T_I^{(k)}$, it can easily generate $T_C^{(k)}(j)$ with $1 \leq j \leq N(k)$, $T_I^{(k+1)}$, $T_C^{(k+1)}(j)$ with $1 \leq j \leq N(k+1)$. However, $N(k+1)$ cannot be predicted, but $N(k+1) = 0$ with very high probability. So in order to let node A receives the packet from node B, it can wake up at time

$$T_{receive}^{(k)} + T + T_I^{(k+1)} - T_I^{(k)}$$

If it receives some signal beginning at time

$$T_{receive}^{(k)} + T + T_I^{(k+1)} - T_I^{(k)} + \sum_{j=1}^{n} T_C^{(k+1)}(j)$$

for some n, where $n = 0$ with very high probability, then this signal is from node B. When the whole packet is received, the node A can turn the radio off and go to sleep. In order to let node A know the random seed $g^{(k)}$ and $N(k)$ of node B in the last period, the random seed $g^{(k)}$ and $N(k)$ should be contained in the kth packet. Once the kth packet cannot be received correctly, by analyzing the $g^{(k-1)}$ and $N(k-1)$ as well as the incoming time of the kth packet, we still can retrieve the value of $g^{(k)}$ and $N(k)$.

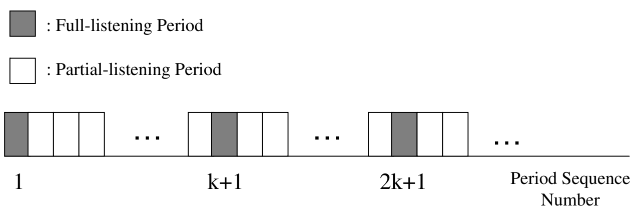

Fig. 5. Period distribution of dominating nodes.

Now, let's consider a dominating node in the network. It may has many neighbors, including dominating neighbors and non-dominating neighbors. Based on the discussion above, for each of the neighbors, the local dominating node can predict the incoming time of the next message from that neighbor. The messages from different neighbors have different incoming time, and the local dominating node needs to be active if and only if there is a neighbor sending a message to it or it is sending messages to other nodes. For other time, it can "sleep" to save energy.

Note that at the beginning, the dominating nodes cannot predict the incoming time of the messages from the neighbors. So that there are one or two periods that the dominating nodes keep listening for incoming messages. We call these periods as full-listening periods, and the others as partial-listening periods. As shown in Fig.5, full-listening periods can appear periodically. By this way, our system can find the insertion of a new node or the great change of some unstable links.

3.4 Implementation

Many literatures have studied the communication properties of WSN, to deal with the challenge of achieving reliability routing in WSN [27]–[29]. In general, communication links in wireless sensor networks are lossy and asymmetric, and some nodes may fail due to some accidents. In this case, it results in unreliable packet delivery.

Different from the ideal case, under unreliable links, the nodes communicate with each other in a probabilistic manner. The packet loss rate from node u to node v can be denoted by $p(u, v)$. In the real world, wireless links are not always symmetric, $p(u, v)$ is not always equal to $p(v, u)$. According to the link estimation method in [29], $p(u, v)$ can be estimated by node v by counting the number of lost packets. In order to know which neighbor is reliable to send messages to the local node v, v can maintain an inbound-neighbor list $N_i(v)$: If there exists node u such that $p(u, v) \geq a$, we add node u into the inbound-neighbor list $N_i(v)$; if $p(u, v) < b$, then we remove node u from the inbound-neighbor list $N_i(v)$. Here, we choose $b < a$ to avoid modifying $N_i(v)$ frequently.

Consider a dominating node v, by listening to incoming messages, it can get the inbound-neighbor list $N_i(v)$, as well as their state and their distance to the sink. In each period, node v will broadcast a packet in the format described in Table 2.

If node v is a non-dominating node or an announcing node, the packet format will be much simpler, as shown in Table 3.

TABLE 2
Packet format from a dominating node v

Packet from a dominating node v.
ID of v
state of v
abstract distance from v to sink
ID of the receiver(destination)
the set of dominating nodes that successfully send messages to v in last period
inbound-neighbor list $N_i(v)$
number of dominating neighbors closer to the sink
random generator seed for CSMA/CA and congestion-delay times
sensing data

TABLE 3
Packet format from a non-dominating node v

Packet from a non-dominating node v.
ID of v
state of v
abstract distance from v to sink
random generator seed for CSMA/CA and congestion-delay times
sensing data

In each period, each node sends a packet out. The first problem is how to determine the parameters in the packet.

- There are several parameters can be easily be determined or obtained, including (1) ID of v, state of v, the abstract distance from v to the sink. (2) The set of dominating nodes that successfully send messages to v in last period. (3) Random generator seed for CSMA/CA.
- During full-listening period, node v may receive messages from several nodes. By monitoring the packet loss from these nodes, the inbound-neighbor list $N_i(v)$ of node v can be obtained and dynamically changed. If $u \in N_i(v)$, then that means u can send messages to node v with high success rate.
- In order to make reliable data delivery, we say that a node u is node v 's neighbor if and only if

$$u \in N_i(v) \quad and \quad v \in N_i(u)$$

By snooping the packets from node u, node v can check whether $v \in N_i(u)$. Further, it can decide whether node u is its neighbor or not. So that node v can maintain a reliable dominating neighbor list. In the scheme, each dominating node selects the dominating neighbor closest to the sink as the relay node, i.e. the receiver. Then, we can get the ID of the receiver. Similarly, we can also know the number of dominating neighbors closer to the sink.

The second problem is that how Reed works based on the received parameters from other nodes.

- The role rotation rules can be checked locally. For the local dominating node v, it maintains a reliable dominating neighbor list. By snooping the messages from the nodes in the list, node v can find whether there is a node who is farther from the sink and violate the condition 1 for the role rotation. Condition 2 can also be checked: for each non-dominating $w \in N_i(v)$, by snooping the message, node v can determine that whether there is a dominating neighbor u of node v such that $w \in N_i(u)$.
- The random generator seed for CSMA/CA and congestion-delay times can be used to help each dominating node to predict the arriving time of incoming messages, then the energy of dominating nodes can be saved by reducing the idle listening time.
- Hop-by-hop retransmission is necessary in data aggregation under unreliable links. Each dominating node stores the fused sensing data during the last period, denoted by $data_k$. By listening to the messages broadcasted from the next-hop node (parent node in the tree), it can determine whether $data_k$ has been received by the next-hop node. If yes, $data_k$ can be deleted. Otherwise, $data_k$ should be fused with the sensing data received in the current period. By this way, it is equivalent with that $data_k$ is retransmitted.

4 Power Consumption

In this section, we mainly discuss the capability of Reed in energy saving, and compare Reed with other energy preserving schemes in data aggregation scenario.

4.1 Approximation Analysis

Let T denote the sampling period, and assume the transmission duration of each packet is T_t. Let n_{dom} (n_{non}) denote the average number of dominating (non-dominating) neighbors of each node, then $n_{non} + n_{dom} = n$ is the average number of neighbors of each node. In general, we have $T \gg nT_t$. And we assume that wireless network is the network is relatively stable, therefore we can ignore the full-listening periods in energy preserving scheme for dominating nodes.

There are four modes for the radio: "Transmit", "Receive", "Idle-listening", and "Sleeping". Referring the existing systems, we have the power consumption of each mode:

$$P_{transmit} > P_{receive} \simeq P_{idle} \gg P_{sleeping} \simeq 0$$

However, turning on or turning off radio too frequently is definitely harmful to circuit, and may reduce the lifetime of sensor nodes. So beside considering power consumption, we also introduce a penalty term E_p for turning on radio each time. Here, we denote E_p as the equivalent energy consumption.

Based on the definition and simplification above, we can approximate the energy consumption of each node in each period. For a non-dominating node, it only needs to turn on radio for one time in each period and keeps in transmission mode for time T_t. Therefore, the energy consumption of a non-dominating node in each period is

$$E_{non} = E_p + T_t P_{transmit}$$

For a dominating node, it needs to wake up for $n = n_{non} + n_{dom}$ times in each period, to receive incoming messages from neighbors. And, it also needs to broadcast a packet. Therefore, the total energy consumption of a dominating node in each period is about

$$E_{dom} = (E_p + T_t P_{receive})n + (E_p + T_t P_{transmit})$$

In Reed, energy consumption is balanced due to role rotation. Therefore, the average energy consumption of each node in a period is about

$$E_{avg} = \frac{n_{dom}E_{dom} + n_{non}E_{non}}{n}$$
$$= (E_p + T_t P_{receive})n_{dom} + (E_p + T_t P_{transmit})$$

So that the average power consumption of each node is about

$$P_{avg} = \frac{E_{avg}}{T}$$
$$= \frac{(E_p + T_t P_{receive})n_{dom} + (E_p + T_t P_{transmit})}{T}$$

In the formula above, $E_p, T_t, P_{receive}, P_{transmit}$ is given and in stationary state n_{dom} is a constant in $[0, 10]$. We can see that P_{avg} is proportional to the inverse of T. So we can reduce the power consumption by increasing the sampling period T. However, the delivery latency of sensing data is proportional to the sampling period T, as a result, increasing T will cause the increase of delivery latency. So there must be a tradeoff between the lifetime of the network and the delivery latency of sensing data.

4.2 Different Versions

In Reed, three techniques are combined to work together to save energy. We want to know that: if we only use one or two techniques in Reed, how much more energy will be used?

Firstly, we don't divide the sensor nodes into two types, i.e. all of the nodes work as dominating nodes, but the energy preserving scheme for dominating nodes is still used. In this case, we have $n_{dom} = n$ and $n_{non} = 0$, the average power consumption of each node is

$$P^{(1)}_{avg} = \frac{(E_p + T_t P_{receive})n + (E_p + T_t P_{transmit})}{T}$$

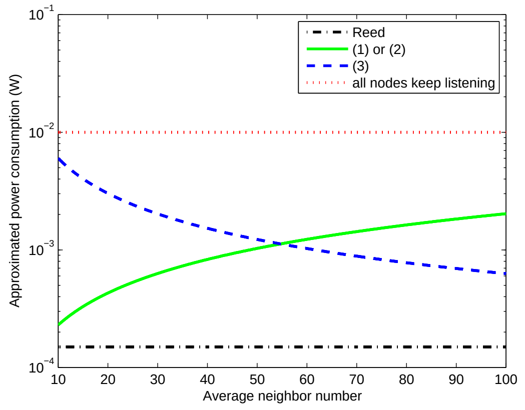

Fig. 6. Comparison of the power consumption of different versions of Reed. In (1), all of the sensor nodes are dominating nodes. In (2), there is no role rotation to balance the energy consumption of dominating nodes and non-dominating nodes, only the power consumption of dominating nodes is considered. In (3), there is no further energy preserving scheme for the dominating nodes.

In the second case, we divide the sensor nodes into two types but no role rotation scheme is used to balance the energy consumption between different types of nodes. In this case, the lifetime of the network depends on the lifetime of the dominating nodes. According to the analysis above, we obtain that the average power consumption of each dominating node is

$$P^{(2)}_{avg} = \frac{(E_p + T_t P_{receive})n + (E_p + T_t P_{transmit})}{T}$$

which is equal to $P^{(1)}_{avg}$.

In the last case, all the sensor nodes are divided into two types and role rotation scheme is used, but no energy preserving scheme is use to reduce the energy consumption of dominating nodes further. In this case, if $P_{receive} \simeq P_{idle}$ and $T \gg T_t$, we know that the average power consumption of each node is about

$$P^{(3)}avg = \frac{E^{(3)}_{avg}}{T}$$
$$= \frac{(TP_{receive})n_{dom} + (E_p + T_t P_{transmit})n_{non}}{nT}$$

In order to simply compare these different versions of Reed, we set $T = 10s$, $T_t = 10ms$, $P_{transmit} = 20mW$, $P_{receive} = P_{listen} = 10mW$ and $E_p = 10^{-4}W \cdot s$. In general, n_{dom} is close to a constant. Here, we set $n_{dom} = 6$. By varying the average neighbor number n, we can get the power consumption of each node (or dominating node) for these different versions of Reed, as shown in Fig. 6. From this figure, we can see that: under the set of parameters above, the power consumption of Reed is about 70 times less than that all the sensor nodes keep listening for incoming messages. Both role rotation scheme and energy preserving scheme for dominating nodes are necessary to reduce the power consumption.

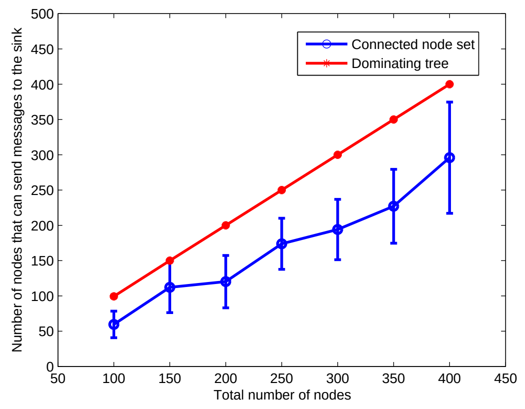

Fig. 7. The number of nodes that can send messages to the sink varies with the total number of the nodes in the network. Assume a connected node set (dominating tree) is constructed and each node forwards messages to its neighbor closest to the sink in the connected node set (dominating tree).

4.3 Comparison with SPAN

SPAN [9] is proposed for multi-hop ad hoc wireless networks to reduce energy consumption without significantly diminished the capacity or connectivity of the network. Here, we want to demonstrate that SPAN cannot be used in data aggregation applications directly.

Firstly, in SPAN a connected node set is maintained. However, this connected node set is dynamically changed due to role rotation among different nodes. In order to find the routes between each node in this connected node set and the sink, in general there are two methods: (1) The sink broadcasts messages to build the routes periodically. However, the connected node set is changed too frequently, so broadcasting will cost a lot of energy. (2) Each coordinator node selects the neighbor closest to the sink to forward messages. However, given a connected node set, where the node density is very small, it is very possible that there exists some nodes cannot find neighbors closer to the sink. Different from SPAN, in Reed each node knows its distance to the sink at the beginning. Assume all the nodes can connect to the sink if we keep them active, then the dominating nodes can connect to the sink along the dominating tree. In Fig. 7, we randomly deploy n nodes in a square of 100×100 meters and deploy the sink at the center of the square, where each node's communication range is 20. Firstly, a connected node set (near optimal) is constructed based on global information. Each node forwards messages to a neighbor in the connected node set, which is the node closest to the sink. Secondly, we construct a dominating tree based on Euclidean distance. Each non-dominating node selects a dominating node to forward messages, and each dominating node forwards messages to its parent in the dominating tree. Fig. 7 calculates the number of nodes that can send messages to the sink using the two methods above. We can see that based on the first method, a large mount of nodes cannot send messages to the sink. But using the dominating tree method, almost all of the nodes can send messages to the sink.

Secondly, in SPAN all the coordinators keep active, that will cost too much energy. Now, we assume the average number of coordinator neighbors of each node $n_{coordinator} \simeq n_{dom}$. Based on the assumption and definition above, we can obtain the average power consumption of SPAN is

$$P_{SPAN} = P_{avg}^{(3)}$$

Compare the average power consumption of Reed P_{avg} and $P_{avg}^{(3)}$ in Fig. 6, we can conclude that SPAN costs much more energy than Reed when $T \gg nT_t$. However, the disadvantage of Reed is that it will introduce more latency into the network. But in most of data aggregation applications, the delivery latency is not the most important issue.

4.4 Comparison with MAC layer protocols

In order to reduce the time of idle listening, several MAC layer protocols have been proposed [30] , such as Sensor-MAC, WiseMAC, and so on. Sensor-MAC (S-MAC) is based on locally managed synchronization and periodic sleep-listen schedules [32]. In S-MAC, nodes coordinate their sleep schedules rather than randomly sleep on their own. Before each node starts its periodic listen and sleep, it needs to choose a schedule and exchange it with its neighbors. If a node receives a schedule from a neighbor before choosing or announcing its own schedule, it sets its schedule to be the same. If two neighboring nodes reside in two different virtual clusters, they follows both of the schedules of these two clusters. In general, the duty cycle of S-MAC is predetermined, therefore the energy

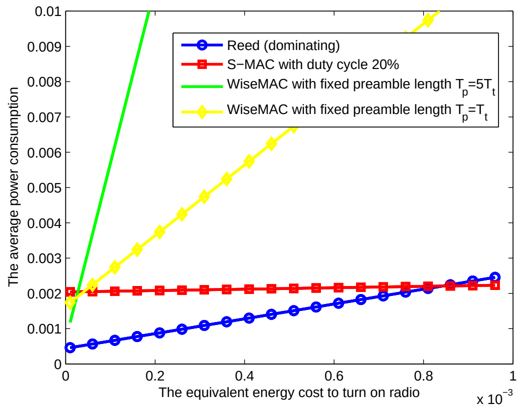

Fig. 8. The approximated power consumption of each node varies with E_p, which is the equivalent energy cost to turn radio on once. In Reed, we only consider the power consumption of dominating nodes. In WiseMAC, in order to simplify the comparison, we assume that the preamble length is fixed.

consumption of S-MAC is close to a constant. WiseMAC combines nonpersistent CSMA with preamble sampling to mitigate idle listening [33]. All nodes in a network sample the medium with the same constant period, but their relative sampling schedule offsets are independent. If a node finds the medium busy, it continues to listen until it receives a data packet or the medium becomes idle again. At the transmitter, each packet has a wake-up preamble of size equal to the sampling period, so that the receiver will be awake when data transmission begins. To reduce the power consumption incurred by the predetermined fixed-length preamble, WiseMAC dynamically determines the length of the preamble. It schedules transmission so that the destination node's sampling time corresponds to the middle of the sender's preamble.

In order to fairly compare Reed with these MAC layer protocols, we only consider the power consumption of dominating nodes in Reed. Similar as above, we still set $T = 10s$, $T_t = 10ms$, $P_{transmit} = 20mW$ and $P_{receive} = P_{listen} = 10mW$ and $n = 30$. In Fig. 8, we compare Reed, S-MAC with duty cycle 20%, and WiseMAC with fixed preamble length. Here, we choose the duty cycle as 20% that is because if the duty cycle become smaller, it will increase the chance of collisions. This figure demonstrates that if the equivalent energy cost to turn on radio E_p is large enough, S-MAC works best among there protocols. But in most of the cases, Reed can save more energy than the other protocols.

5 Communication Reliability

In this section, we use our discrete-time event simulator based on Java to simulate the schemes proposed. Simulation shows that Reed can work well under unreliable wireless links.

5.1 Simulation Environment and Scenario

In wireless sensor networks, it often assumes that the receiving power P_r depends on the transmitted power P_t and the distance between two nodes d:

$$P_r \propto \frac{P_t}{d^n}$$

where n is called the path-loss exponent and the typical values for n range from 2 (free space) to 4 (indoor). Now, we set n as 3.5 in our simulator.

For given two nodes, some packets transferring between them might be lost while some others might be successfully received. For this probabilistic behavior a function called *packet error rate* is obtained [26]. Given SIR(Signal to Noise Rate), this function gives the probability of losing a packet. Assume the local node is receiving signals from node $N_1, N_2, ..., N_k (k > 1)$, then for the signal from node N_i, the SIR is

$$SIR_i = \frac{P_{ri}}{N_0 + \sum_{j \neq i} P_{rj}}$$

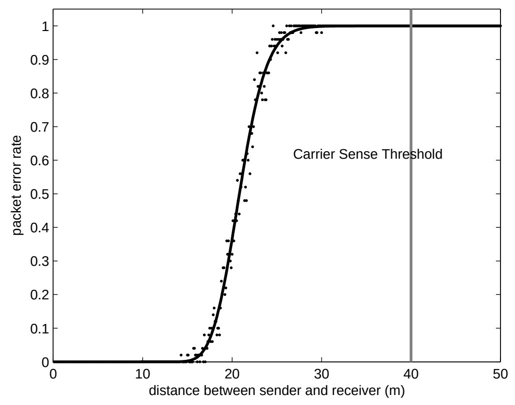

Fig. 9. The relationship between packet error rate and the distance between two nodes, without the inference from other nodes.

where P_{ri} is the receiving power of the signal from node N_i and N_0 is the power of environmental noise.

Fig.9 shows the channel model used in our simulator when there is no other nodes' interference. We notice that there exists a carrier sense threshold P_{cs}. When a packet is transmitting to a node and the receiving power of the node $P_r > P_{cs}$, the receiver can observe the channel busy. The channel model used in our simulator is similar to the experimental results in [27] and [28].

In our simulation, we randomly deploy n nodes into an $100 \times 100m^2$ square. These nodes are uniformly distributed and the sink is deployed at the center of the square. We assume that each node knows its Euclidean distance to the sink, therefore each node selects the neighbor closest to the sink to forward messages. Let T denote the sampling period, which is set $T = 10s$ in this simulation. For the role rotation of node u, we set the duration of an announcing state is $2T$, $T_a = (1 + R)100T$ and $T_c = (1 + 3N(u)R)100T$. Since links are not reliable, all estimation for link quality is based on the received messages during last 10 periods.

In our simulation, we let all the nodes stay in dominating state at the beginning, and run our system for 10000 seconds.

5.2 Number of Dominating Nodes

Dominating nodes cost much more energy than non-dominating nodes, so we hope that the number of dominating nodes should be as a few as possible, while a dominating tree is maintained. Let's consider the number of dominating nodes required to form a hexagonal grid layout, where a dominating node is placed at each vertex of a hexagon. According to literature [9], we know that for an square of 100^2 meters and communication range of 20 meters, the number of required dominating nodes is $C_{hex} = 19$. Although the hex grid layout may not be optimal, it produced a connected backbone with very a few dominating nodes.

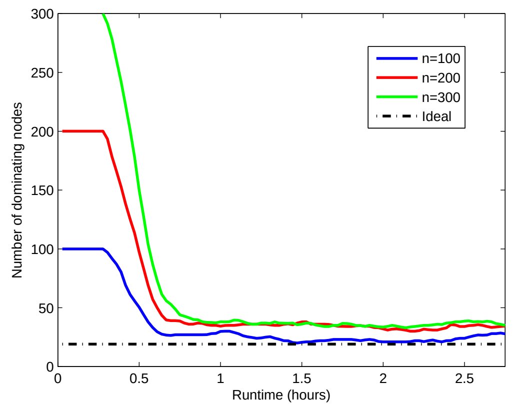

Fig. 10. The number of dominating nodes varies with the simulation runtime under different node density. The curves are smoothened.

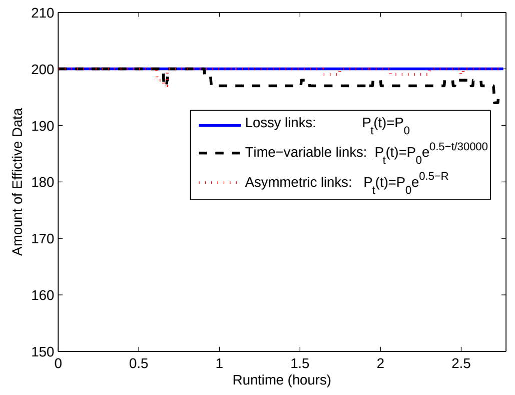

Fig. 12. The amount of effective data at the sink varies with the simulation runtime under different kinds of links.

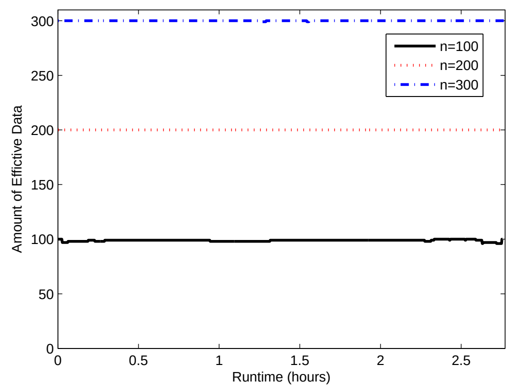

Fig. 11. The amount of effective data at the sink varies with the simulation runtime with different node density.

Fig. 10 shows the number of dominating nodes varies with the simulation runtime when different numbers (100,200,300) of nodes are deployed in a 100×100 m^2 square area. At the beginning all of the nodes stay in dominating states. As the time goes on, the number of dominating nodes can be dynamically reduced. This figure implies that as the node density increases, the density of dominating nodes keeps close to a constant. Comparing with the ideal case of hex grid layout, Reed selects a little more dominating nodes since all decisions are made locally (using one-hop information).

5.3 Reliability

In this section, we try to analyze the reliability of Reed under lossy links. In our simulation, the sink stores all the most recent received sensing data from different sources. Assume that the sensing data from source u, stored at the sink, was sensed at time $T_{sensing}(u)$. The the delivery latency of the sensing data from source u is $T_{current} - T_{sensing}(u)$, where $T_{current}$ is the current time. If the delivery latency of the sensing data from one source is more than 100 seconds, we call this sensing data as un-effective sensing data that may imply the breaking of some communication links. So, we can use the amount of effective sensing data at the sink to estimate the reliability of the network. Fig.11 shows the amount of effective sensing data varies with the runtime. We find that most of the sensing data stored in the sink is effective, that means Reed is reliable under lossy links.

In real applications, the quality of wireless links strongly depends on several parameters, including distance between sender and receiver, altitude of sender or receiver from the ground, humidity of the environment, physical layer coding schemes and antenna orientation of sender or receiver. In a network, different nodes may have different communication distance due to the factors above, and it is very possible that one node can send messages to its neighbor, but the neighbor can not send messages back, so the links are asymmetric. Since environment may change from time to time, for example we cannot keep humidity of the environment at a constant, the quality of wireless links may also change with time. Now we evaluate Reed under asymmetric lossy links and time-variable lossy links. Under asymmetric lossy links, we assume that different nodes have different transmission power, which is $p_t = p_0 e^{0.5-R}$, where p_0 is a constant power and R is a random variable uniformly distributed in $[0, 1]$. R is different for different nodes. Under time-variable lossy links, we let the communication range decrease with the time, so we set the transmission power as $p_t(t) = p_0 e^{0.5-t/30000}$. Therefore, the transmission power varies from $p_0 e^{0.5}$ to $p_0 e^{-0.5}$.

Fig.12 shows the amount of effective sensing data at the sink varies with the simulation runtime under different types of links. We can conclude that Reed have high reliability under all the links in our simulation. However, among these links, the time-variable links have the biggest impact to the reliability of the system.

6 Conclusions

This paper proposed a reliable and energy-efficient data aggregation protocol Reed, in which each sensor node periodically generates sensing data and all these sensing data is aggregated to the sink. In Reed, the power consumption of each node is significantly reduced by minimizing the idle-listening time of radio. Comparing with other existing methods or systems, Reed can work well in distributed multi-hop sensor networks.

Analysis and simulations show that Reed can significantly extend the network lifetime that the energy consumption of the system with Reed is about 70 times less than the network that radios keep active if the sampling period is set as 10 seconds. At the same time, high reliability of the network is ensured, even if the communication links is lossy, asymmetric and time-variable.

Acknowledgments

The research presented in this paper is supported in part by the National Natural Science Foundation (60736027, 60574087), 863 High Tech Development Plan (2007AA01Z475, 2007AA04Z154, 2007AA01Z480, 2007AA01Z464) and 111 International Collaboration Program, of China.

References

[1] V. Shnayder, M. Hempstead, et al. ”Simulating the power consumption of large-scale sensor network applications.” Proceedings of the Second International Conference on Embedded Networked Sensor Systems (SenSys), Baltimore, MD, United States, 2004. 188-200.

[2] S. C. Ergen and P. Varaiya. ”PEDAMACS: Power efficient and delay aware medium access protocol for sensor networks.” IEEE Transactions on Mobile Computing 5(7): (2006), 920-930.

[3] R. Szewczyk, J. Polastre, et al. ”Lessons from a sensor network expedition.” Proceedings of the First European Workshop on Wireless Sensor Networks (EWSN), January 2004.

[4] J. Carle and D. Simplot-Ryl. ”Energy-Efficient Area Monitoring for Sensor Networks.” Computer 37(2): (2004), 40-46.

[5] Y. Xu, S. Bien, Y. Mori, et al. ”Topology Control Protocols to Conserve Energy in Wireless Ad Hoc Networks.” January 2003, CENS Technical Report 0006.

[6] Y. Xu, J. Heidemann, and D. Estrin. ”Adaptive energy-conserving routing for multihop ad hoc networks.” Technical Report TR-2000-527, USC/Information Sciences Institute, Oct. 2000. Available at ftp://ftp.isi.edu/isi-pubs/tr-527.pdf.

[7] P. Gupta and P. R. Kumar. ”Critical power for asymptotic connectivity.” Proceedings of the IEEE Conference on Decision and Control 1: (1998), 1106-1110.

[8] F. Xue and P. R. Kumar. ”The Number of Neighbors Needed for Connectivity of Wireless Networks.” Wireless Networks 10(2): (2004), 169-181.

[9] B. Chen, K. Jamieson, et al. ”Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks.” Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM, Rome, 2001. 85-96.

[10] S. Guha and S. Khuller. ”Approximation algorithms for connected dominating sets.” Lecture Notes in Computer Science, Barcelona, Spain, 1996. 179

[11] J. Wu and H. Li. ”On Calculating Connected Dominating Sets for Efficient Routing in Ad Hoc Wireless Networks,” Proc. Third ACM Int’l Workshop Discrete Algorithms and Methods for Mobile Computing and Comm., 1999.

[12] F. Dai and J. Wu. ”Distributed dominant pruning in Ad Hoc networks.” IEEE International Conference on Communications, Anchorage, AK, United States,2003. 353-357.

[13] I. Stojmenovic, M. Seddigh, et al. ”Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks.” IEEE Transactions on Parallel and Distributed Systems 13(1): (2002), 14-25.

[14] J. Wu, F. Dai, et al. ”Iterative local solutions for connected dominating set in ad hoc wireless networks.” 2nd IEEE International Conference on Mobile Ad-hoc and Sensor Systems, MASS 2005, Washington, United States, 2005. 581-588.

[15] M. Ma, M. Chi, et al. ”A cross-layer data gathering scheme for heterogeneous sensor networks based on polling and load-balancing.” IEEE Wireless Communications and Networking Conference, WCNC, Kowloon, China, 2007. 3303-3308.

[16] Crossbow sensor system, http://www.xbow.com

[17] BTNODES sesnor system, http://www.btnode.ethz.ch

[18] TMote Sky sensor system, http://www.moteiv.com

[19] R. Cristescu, B. Beferull-Lozano, et al. ”On network correlated data gathering.” Proceedings of IEEE INFOCOM, Hongkong, China, 2004. 2571-2582.

[20] R. Cristescu, B. Beferull-Lozano, et al. ”Network correlated data gathering with explicit communication: NP-completeness and algorithms.” IEEE/ACM Transactions on Networking 14(1): (2006), 41-54.

[21] K. Yuen, B. Liang, et al. ”A distributed framework for correlated data gathering in sensor networks.” IEEE Transactions on Vehicular Technology 57(1): (2008), 578-593.

[22] C. Hua and T.-S. P. Yum. ”Asynchronous random sleeping for sensor networks.” ACM Transactions on Sensor Networks 3(3): (2007), 1267063.

[23] C. Liu, K. Wu, et al. ”Random coverage with guaranteed connectivity: Joint scheduling for wireless sensor networks.” IEEE Transactions on Parallel and Distributed Systems 17(6): (2006), 562-575.

[24] S. Ren, Q. Li, et al. ”Design and analysis of sensing scheduling algorithms under partial coverage for object detection in sensor networks.” IEEE Transactions on Parallel and Distributed Systems 18(3): (2007), 334-350.

[25] D. Tian and N. D. Georganas. ”A coverage-preserving node scheduling scheme for large wireless sensor networks.” Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications, Atlanta, GA, United States, 2002. 32-41.

[26] J.-M. Dricot and P. De Doncker. ”High-accuracy physical layer model for wireless network simulations in NS-2.” 2004 International Workshop on Wireless Ad-Hoc Networks, Oulu, Finland, 2005. 249-253.

[27] G. Anastasia, E. Borgiaa, et al. ”Understanding the real behavior of Mote and 802.11 ad hoc networks: an experimental approach.” Pervasive and Mobile Computing 1 (2005) 237C256.

[28] J. Zhao and R. Govindan. ”Understanding packet delivery performance in dense wireless sensor.” Proceedings of the First International Conference on Embedded Networked Sensor Systems, SenSys, Los Angeles, CA, United States, 2003. 1-13.

[29] A. Woo, T. Tong, et al. ”Taming the underlying challenges of reliable multihop routing in sensor networks.” Proceedings of the First International Conference on Embedded Networked Sensor Systems, SenSys, Los Angeles, CA, United States, 2003. 14-27.

[30] I. Demirkol, C. Ersoy, and F. Alagöz. ”MAC Protocols for Wireless Sensor Networks: A Survery.” IEEE Communications Magazine, April 2006. 115-121.

[31] M. Buettner, G. V. Yee, et al. ”X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks.” Proceedings of the Fourth International Conference on Embedded Networked Sensor Systems, SenSys, Boulder, CO, United States, 2006. 307-320.

[32] W. Ye, J. Heidemann, and D. Estrin, ”Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks.” IEEE/ACM Trans. Net., vol. 12, no. 3, June 2004, 493-506.

[33] C. C. Enz, A. El-Hoiydi et al., ”WiseNET: An Ultralow-Power Wireless Sensor Network Solution.” IEEE Comp., vol. 37, no. 8, Aug 2004, 62-70.

Hongchao Zhou received the B.S. degree in mathematics and physics, the M.S. degree in control science and engineering from Tsinghua University, Beijing, China, in 2006 and 2008, respectively. He received the M.S. degree in electrical engineering from California Institute Technology, Pasadena, in 2009. Currently, He is a Ph.D. candidate in electrical engineering, California Institute of Technology.

His working experience includes a research internship at IBM China Research Lab, from Jan 2006 to Jun 2008. His research interests include information theory, stochastic systems, wireless networks.

Xiaohong Guan (S'89-M'93-SM'94-F'07) received the B.S. and M.S. degrees in automatic control from Tsinghua University, Beijing, China, in 1982 and 1985, respectively, and the Ph.D. degree in electrical engineering from the University of Connecticut, Storrs, in 1993.
From 1993 to 1995, he was a Consulting Engineer at PG&E. From 1985 to 1988, he was with the Systems Engineering Institute, Xian Jiaotong University, Xian, China. From January 1999 to February 2000, he was with the Division of Engineering and Applied Science, Harvard University, Cambridge,MA. Since 1995, he has been with the Systems Engineering Institute, Xian Jiaotong University, where he is also currently a Cheung Kong Professor of systems engineering, the Director of the State Key Laboratory for Manufacturing Systems, and the Director of the Systems Engineering Institute. His research interests include allocation and scheduling of complex networked resources, network security, and sensor networks.