A Better Choice for Sensor Sleeping

Ou Yang and Wendi Heinzelman

Dept. of ECE, University of Rochester, Rochester, NY, 14620, USA
oyang@ece.rochester.edu,wheinzel@ece.rochester.edu

Abstract. *Sensor sleeping is a widely-used and cost-effective technique to save energy in wireless sensor networks. Protocols at different stack levels can, either individually or simultaneously, make the sensor sleep so as to extend the application lifetime. To determine the best choice for sensor sleeping under different network conditions and application requirements, we investigate single layer and multi-layer sleeping schemes at the routing and MAC layers. Our results show that routing layer sleeping performs better when there is high network redundancy or high contention, while MAC layer sleeping performs better when there is low contention or in small networks. Moreover, multi-layer sleeping requires cross-layer coordination to outperform single layer sleeping under low contention. Therefore, our conclusions can not only guide the implementation of practical sensor networks, but they also provide hints to the design of cross-layer power management to dynamically choose the best sleeping scheme under different network and application scenarios.*

Key words: Sensor sleeping, Directed Diffusion, SMAC, Cross-layer coordination

1 Introduction

Wireless sensor networks are gaining increasing attention for practical uses ranging from military surveillance to civil monitoring due to their low cost, ease of deployment and good coverage of the monitored area. However, one of the main issues preventing the ubiquitous use of wireless sensor networks is how to support an application for extended periods of time, as sensors are usually battery-powered and hence highly constrained in energy supply. Consequently, to save energy, the idea of "making sensors sleep when they are not used" has motivated much research in wireless sensor networks [1] [2] [3] [4] [5] [6] [7]. In particular, the selection of source node(s) that should transmit data to the sink(s), which is done at the application layer, allows redundant source nodes to sleep in order to save energy for later use. Topology control, which is usually performed at the network layer, creates a routing backbone for data delivery so that the remaining non-router nodes can sleep. Turning off routing nodes that are not directly involved in the delivery of data can also be done by the routing protocol. Finally, duty cycling is often employed by the MAC protocol, allowing sensors to sleep periodically to reduce their idle listening, which is energy intensive.

Since protocols at different stack layers can make sensors sleep, the question becomes at which layer should sleeping be used in order to extend application lifetime for a specific network and application scenario? In other words, is there a particular layer in which sleep scheduling provides the most benefit? Can sleep scheduling be done at multiple layers independently for even more gain than simply sleeping at a single layer? Or is some sort of cross-layer coordination of sleep scheduling needed to obtain the maximum benefit in terms of extending application lifetime? To examine these questions, in this paper we compare sleeping techniques employed by routing protocols and MAC protocols under various network scenarios and application requirements. Application layer sleeping schemes are currently not considered to avoid any application-specific conclusions. Thus, the contributions of this paper are:

(1) We propose a sleeping scheme for Directed Diffusion [9], which significantly improves the application lifetime by allowing nodes not involved in the transmission of data to sleep periodically. (2) We provide detailed performance comparisons of MAC layer sleeping and routing layer sleeping, under various node densities, different network scales, diverse numbers of source nodes and varying application data rates. These results show that MAC layer sleeping works better under conditions of low contention, while routing layer sleeping works better in networks with high redundancy. (3) We also examine the performance of making sensors sleep at the routing and MAC layers simultaneously, with and without cross-layer coordination. Our preliminary results show that it is necessary to employ cross-layer coordination between the layers to obtain further improvement in throughput when the contention in the network is low. (4) Our conclusions can be used to design a policy for a centralized power manager that dynamically decides which layer(s) should have control over sleep scheduling as network parameters vary or application requirements change over time.

2 Motivation and Background

Wireless sensors have stringent energy constraints since they are usually battery-powered, and oftentimes it is impractical to recharge the sensors manually due to their large-scale deployment. Hence, it is critical to employ energy-saving techniques so as to support an application for extended periods of time. Among the existing energy-saving strategies, e.g., flooding avoidance [8] [9], traffic balancing using cost functions [10] [11], data fusion [12], etc., putting sensors into the sleep state is the most widely-used and cost-effective way to prolong the application lifetime. By turning off the radio of a sensor at appropriate times, various sleeping schemes [1] [2] [3] [4] [5] [6] [7] aim to cut down on "idle listening", which in wireless sensor networks provides little benefit yet consumes almost as much power as transmitting or receiving data [13].

To save the most energy, it is intuitive to make sensors sleep whenever possible. This implies that (1) redundant source nodes may sleep when their sensed data is not required by the sink, (2) redundant routing nodes may sleep when they have no data to relay, and (3) source nodes and routing nodes that are

involved in the data delivery may also sleep when it is not their turn to transmit or receive data. Correspondingly, existing sleeping schemes are implemented in such a way that different layers perform these different tasks mentioned above.

Specifically, the application layer selects and activates only some of the source nodes to reduce energy drain while maintaining desired QoS [1]. For example, in the application of target tracking [1], all sensors within a certain distance of the target could be source nodes, but only a few are finally activated by the application layer to achieve low distortion and reduce the energy consumption.

Routing nodes, on the other hand, are usually selected and activated by the network layer in two ways. The first way is called topology control [2] [3], which aims to establish a routing backbone to guarantee the network connectivity, and consequently non-backbone nodes can sleep. Topology control also updates the backbone periodically according to the remaining energy of each sensor to balance the energy consumption. For instance, GAF [2] is a topology control protocol, which divides a network into virtual grids with only one sensor in each grid activated at any time, constituting a backbone network to route all the data. Topology control is often used in large scale, dense networks where many nodes provide redundant routes. Hence, making all of them except the activated sensors sleep can significantly extend the application lifetime.

The second way of making sensors sleep at the network layer is through routing protocols. Routing protocols [4] [5] update routes periodically, and save energy by turning off the routing nodes that are not involved in the data delivery. For example, the technique described in [4] puts sensors to sleep by monitoring routing control messages and data transmissions so that only useful routers are kept active. The technique described in [5] also utilizes the routing decisions so that sensor nodes do not wake up when they are not part of a routing path. Compared to topology control, those routing protocols with sleeping do not waste the energy of non-used routing nodes, and they are also suitable for sparse or not large scale networks where topology control performs poorly.

Once source nodes and routing paths are determined, the MAC layer can put sensors into sleep-awake cycles [6] [7] so that idle-listening can be further reduced due to the fact that the traffic load in wireless sensor networks is usually not very high [6]. For example, SMAC [6] makes each sensor broadcast its sleep-awake schedule once in a while, so that its neighboring sensors can hear it and synchronize their schedules with it. Once sensors are synchronized, they use RTS/CTS/DATA/ACK sequences to communicate during the awake period of each cycle. BMAC [7], using periodic low power listening, forces a sensor to sleep if nothing is detected on the media in the sensor's awake duration. However, the benefit of longer application lifetime provided by MAC layer sleeping accompanies lower throughput and higher latency.

As sleeping can be implemented at the application layer, network layer and MAC layer individually, questions arise as to which layer provides the most benefit for determining when a sensor should sleep, under what conditions should each sleeping scheme be chosen, and whether cross-layer coordination is needed to obtain further improvement when sleeping schemes are employed at different

layers simultaneously. The answer to these questions may not only guide us in the implementation of practical sensor networks, but they may also provide hints into the design of cross-layer power management.

3 Sleeping at Different Layers Individually and Simultaneously

As wireless sensing applications operate in diverse networking environments and have a range of QoS requirements, source node selection at the application layer is always application-specific, and hence it is hard to generalize its performance. Therefore, to avoid any application-specific conclusions, in this paper we choose a general query-based application, and we focus on more general sleeping strategies at the network and MAC layers. Specifically, we look into sleeping schemes implemented in the routing and MAC protocols, defined as "routing layer sleeping" and "MAC layer sleeping", respectively, throughout this paper. Topology control (done by the network layer) is currently not investigated. We make conclusions by comparing the performance of (1) a non-sleeping routing protocol with a non-sleeping MAC protocol, (2) a non-sleeping routing protocol with a sleeping MAC protocol, (3) a sleeping routing protocol with a non-sleeping MAC protocol, and (4) a sleeping routing protocol with a sleeping MAC protocol.

For the non-sleeping routing protocol, we choose Directed Diffusion [9], as it is specially designed for data-centric sensor networks and widely accepted as a typical routing protocol for wireless sensor networks. However, Directed Diffusion does not include any sleeping strategy. We, therefore, propose an improved Directed Diffusion with sleeping as our sleeping routing protocol (see details in Section 3.1). The sleeping Directed Diffusion performs exactly the same as the original Directed Diffusion in terms of source node discovery and routing path establishment and maintenance, but sleeping Directed Diffusion allows routing nodes to sleep during the interval between two successive routing updates, if they are not reinforced by the sink to relay data. Therefore, the application lifetime may be prolonged by getting data from another routing path once the previous path dies due to energy depletion.

For the MAC layer, we use IEEE 802.11 and SMAC as the non-sleeping and sleeping protocols, respectively. SMAC is a typical MAC protocol with duty cycled sleeping for wireless sensor networks. Except for the duty cycle, SMAC is similar to IEEE 802.11. In other words, SMAC with 100% duty cycle has the same performance as IEEE 802.11 under light traffic load. Hence, comparing the performance of the network using IEEE 802.11 and SMAC provides clear insight into the benefits and drawbacks of duty cycling the sensors at the MAC layer.

3.1 Sleeping at the Routing Layer

To understand our proposed sleeping Directed Diffusion, we first review the mechanisms of Directed Diffusion, and then we explain our implementation of sleep-awake cycles used with Directed Diffusion.

Directed Diffusion. Directed Diffusion [9] includes two phases, an exploratory phase and a reinforcement phase, which together allow a sink node to obtain desirable data from source nodes. The exploratory phase is used to discover data sources at a low data rate, while the reinforcement phase is used to pull down the desirable data at a high data rate.

In the exploratory phase, a sink starts broadcasting INTEREST packets periodically in the network. An INTEREST packet includes a description of desired data attributes and indicates the exploratory rate of this specific data (usually as low as one packet per hundreds of seconds). A node that receives an INTEREST packet floods it to all of its neighbors. Meanwhile, every node maintains a gradient table, caching each distinguishable INTEREST packet in terms of where it came from (previous hop) and what the desired data rate is as the local routing information. Each caching entry is called a gradient, which expires and hence will be removed from the gradient table after a certain time, so as to take care of topology changes or node failures. New gradients will be established by the periodic INTEREST packets flooded in the network. As a result, after some time, all the nodes, including the data sources in the network know the INTEREST, and all possible routing paths are established in a distributed manner by checking gradient information at each node. When a source node receives an INTEREST packet, it broadcasts DATA packets to all its neighbors at the exploratory rate. All the intermediate nodes cache and flood all distinguishable DATA packets and discard duplicate ones (e.g., if the time stamp is very close - less than the interval of two successive high rate DATA packets - to any received DATA packet). Finally, the sink node receives the DATA packets that it is interested in from multiple paths. Then the sink starts the reinforcement phase.

In the reinforcement phase, the sink node has a specific policy to reinforce some of the routing paths to pull down the data from the source nodes at high data rates. The policy in our experiments is to reinforce the neighbor that delivered the exploratory data first. The sink node unicasts a positive reinforcement packet, which is actually an INTEREST packet with high desired data rate (usually tens of times higher than the exploratory rate), to the selected neighbor, and then the selected neighbor forwards this positive reinforcement packet to the next hop, which is chosen by the same policy. When a source node receives a positive reinforcement packet, it starts sending back DATA packets at the requested high data rate, along the path where the positive reinforcement packet came from. Therefore, the sink node finally obtains data at the desired high rate from the reinforced path. In the case of topology changes, node failures or link quality changes, a new path may be positively reinforced. Hence, every sensor periodically checks if a negative reinforcement is needed to slow down the data delivery on the previously reinforced paths.

Directed Diffusion does not allow sensors to sleep. However, not every routing node is involved in the data delivery all the time, as only the least delayed routing path is positively reinforced. Hence, allowing redundant routing nodes to sleep

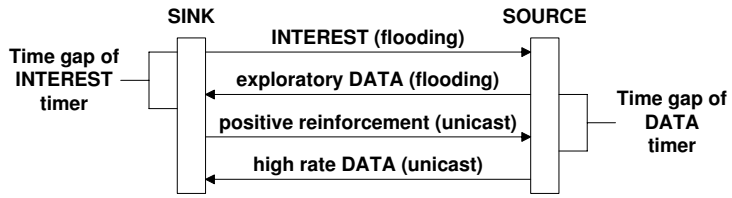

Fig. 1. Time sequences of path establishment in sleeping Directed Diffusion.

may save other routing paths and contribute to a longer application lifetime when the previous ones run out of energy.

Sleeping Directed Diffusion. To make routing nodes sleep when they are not involved in the data delivery, it is necessary to note the importance of INTEREST flooding and exploratory DATA flooding, and figure out the time sequences of path establishment and maintenance in Directed Diffusion.

As mentioned above, in Directed Diffusion there are two critical floodings throughout the network. One is the periodic flooding of an INTEREST packet, initiated by the sink node. All routing nodes need to be awake to forward this packet so that source nodes can finally receive it, and a gradient table can be established to route the DATA packets that are going to follow. The other flooding is exploratory DATA flooding, periodically initiated by a source node. All routing nodes need to be awake to forward those packets so that the sink node can finally receive them and start positive reinforcement based on their arrival times. As a result, for each distinguishable INTEREST (query for different data attributes), we define two timers, INTEREST timer and DATA timer, at each node, for INTEREST flooding and exploratory DATA flooding, respectively. The two timers are scheduled and fire periodically. Specifically, INTEREST timer is scheduled after an INTEREST flooding ends, and fires before the next INTEREST flooding starts, while DATA timer is scheduled after an exploratory DATA flooding ends, and fires before the next exploratory DATA flooding starts. A sensor may sleep when the two timers are pending, but it MUST wake up once either timer fires, and remain awake until the timer is rescheduled.

There is a time gap between when a timer fires and when it is rescheduled. The time gap is used to wait for the response of the corresponding flooding so as to establish a reinforced path from the sink node to the source node. As shown in Fig. 1, path establishment starts with INTEREST flooding, followed by the exploratory DATA flooding, which is then followed by positive reinforcement unicasting, and consequently followed by high rate DATA unicasting. Hence, the time gap of the INTEREST timer is used to wait for the exploratory DATA flooding if new source nodes are discovered, while the time gap of the DATA timer is used to wait for the positive reinforcement packets so that one of the source nodes can be reinforced and start delivering high rate data.

If a node does not receive any exploratory DATA packet during the time gap of its INTEREST timer as a result of no source node available or packet loss, the exploratory DATA timer is not initiated. Hence, the node follows the INTEREST timer to sleep (when it is pending) and wake up (when it expires). Otherwise, the exploratory DATA timer is initiated, and it is scheduled periodically according to the exploratory rate. If the node receives a positive reinforcement packet during the time gap of its DATA timer, the node is located on the reinforced path (involved in the data delivery), and thus cannot sleep until it is negatively reinforced or it runs out of energy. If the node does not receive any positive reinforcement packet during the time gap of its DATA timer, the node goes to sleep when both of the timers are pending (no flooding is going on), and wakes up whenever at least one of the timers expires (at least one flooding is going on).

3.2 Sleeping at the MAC Layer

SMAC is a MAC protocol explicitly proposed for wireless sensor networks, with reducing energy consumption as its primary goal [6]. It introduces periodic sleep-awake cycles for each node to eliminate idle-listening. Used for transmitting and receiving data, the awake period of a cycle has a fixed length, which is determined by the physical layer and MAC layer parameters. The sleeping period of a cycle, instead, can be set longer or shorter, which influences the power consumption of SMAC, as well as the latency incurred by sleeping. Hence, variations in duty cycle, which is defined as the ratio of the awake period to a complete sleep and awake cycle, leads to corresponding variations in the performance of SMAC.

For the convenience of communication, and to reduce the control overhead, SMAC tries to synchronize every node, so that nodes sleep and wake up simultaneously. To achieve this, every node periodically broadcasts its schedule in a SYNC packet, so that neighbors who hear the SYNC packet start following the same schedule. However, some nodes may not hear the SYNC packets from their neighbors because they have already been running different schedules. Hence, every node must keep awake for a whole SYNC packet interval once in a while, so that different schedules of its neighbors can be heard. A node that has a different schedule from its neighbors may follow both schedules at the same time.

During the awake time, SMAC is similar to IEEE 802.11, which (1) uses RTS/CTS to solve the hidden terminal problem, (2) uses physical carrier sensing and virtual carrier sensing to avoid collision, and (3) uses RTS/CTS/DATA/ACK sequences to guarantee successful unicast transmissions. If a node fails to access the media, it goes back to sleep until the next awake period. If, on the other hand, a node successfully accesses the media, it does not sleep until it finishes the current transmission.

3.3 Sleeping at Both Routing and MAC Layers

To employ sleeping Directed Diffusion and SMAC simultaneously, we can either simply implement them as they are, at the routing layer and the MAC layer,

respectively, or do cross-layer coordination between the routing layer and the MAC layer to improve the overall performance.

There are various ways to coordinate the routing and MAC layer sleeping. We implement two methods in our simulations. The first cross-layer coordination is based on priority. As sleeping Directed Diffusion puts all the nodes that are not involved in the data delivery to sleep during successive floodings, there is no need to make those nodes wake up at the MAC layer according to its duty cycle, as no data needs to be transmitted. Hence, sleeping Directed Diffusion should have higher priority than SMAC to schedule the nodes. In other words, SMAC only effectively schedules a node when sleeping Directed Diffusion needs to keep this node active. The second cross-layer coordination is differentiation between routing layer sleeping and energy depletion. As SMAC updates a neighbor list at each node by recognizing SYNC packets sent by its neighbors for a given period of time, long sleeping time of a node scheduled by sleeping Directed Diffusion may make the node's neighbors mistakenly drop its information from their neighbor lists, as no SYNC packet is sent from this node during its sleeping time. We, therefore, make a SYNC packet bear the remaining energy of its sender, so that the receiving nodes can easily tell the status of the sender, and remove the sender from its neighbor list only when it is running out of energy.

4 Simulations and Discussions

In this section, we analyze the pros and cons of making sensors sleep at individual layers under different network scenarios and application requirements. We also give some preliminary results on the necessity of cross-layer coordination when sensors sleep at both layers. Two metrics are considered. The first metric is throughput, which is the total number of packets received by the sink node. In general, the higher the throughput, the more information is collected at the sink, which corresponds to a longer application lifetime. However, a high throughput does not necessarily imply a good data delivery ratio. Therefore, we define the second metric, data delivery ratio, as the throughput divided by the number of packets the sink node should ideally receive. Ideally, the sink node receives as many packets as generated by a single reinforced source node during the whole data delivery period. The higher the data delivery ratio is, the fewer packets are lost. In general, for a given application and network deployment, we desire both throughput and data delivery ratio to be high.

We use ns-2.32 [14] to simulate the performance of all combinations of a non-sleeping/sleeping routing protocol (Directed Diffusion/sleeping Directed Diffusion) and a non-sleeping/sleeping MAC protocol(IEEE 802.11/SMAC[1]). We assume that (1) sensors are static and randomly distributed in a given area, (2) there is only one sink node and more than one source node, (3) the sink node has infinite power supply, while other nodes have 22J initial energy, (4) each node's power consumptions in transmitting, receiving, and idle status are set the same

[1] Adaptive listening [9] is used in the simulation of SMAC.

at 50mW, based on measurements of CC1000, a radio chip for MICA2 motes, and CC2420, a radio chip for IEEE 802.15.4 [13], (5) the size of an application layer data packet is 64 bytes, (6) MAC layer bandwidth is 2Mbps, and (7) the communication range of a sensor is 250m.

For Directed Diffusion, we assume that the sink node floods an INTEREST packet every 30s. Each gradient entry in the gradient table is valid for 50s. The exploratory rate is 1 packet per 100s. Negative reinforcement check is executed every 6 high rate DATA packet intervals. For sleeping Directed Diffusion, either timer has a 6s time gap. For SMAC, the size of a DATA packet is 50 bytes, the size of a SYNC packet is 9 bytes, and the size of other control packets, like RTS, CTS, and ACK, are 10 bytes. A SYNC packet is sent by each node every 10 duty cycles.

We first look into the performance of individual layer sleeping schemes, namely sleeping Directed Diffusion and SMAC, and then compare their performances under different situations. Finally, we show the performance differences of sleeping at both routing and MAC layers, with and without cross-layer coordination. By default, 30 nodes are randomly distributed in an 800mX800m area. There is 1 sink node and 5 source nodes. A reinforced source node generates 3 application layer packets per 10s. For each scenario, 10 topologies are generated, and the results are averaged over 20 simulations, except as noted.

4.1 Performance of Single Layer Sleeping

Fig. 2 shows the throughput of single layer sleeping schemes over time in one simulation. IEEE 802.11 is used as the common MAC protocol in the simulation of sleeping Directed Diffusion, while Directed Diffusion is used as the common routing protocol in the simulation of SMAC. As we can see, both sleeping Directed Diffusion and SMAC can significantly improve the data delivery period, and hence the overall throughput. However, sleeping Directed Diffusion has a "QoS pause" between 740s to 800s, as the throughput remains the same during this period of time. Since a new path reinforcement is always triggered by an exploratory DATA flooding, there might be a gap during which the old path has

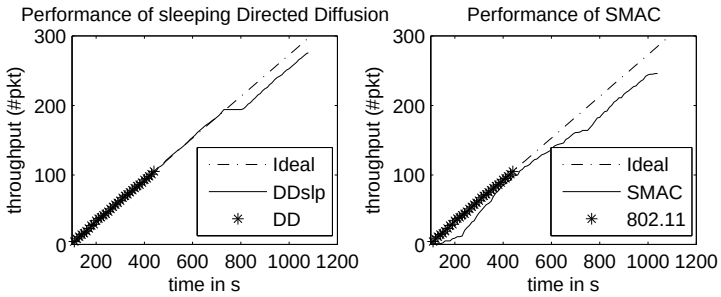

Fig. 2. Performance of sleeping Directed Diffusion and SMAC.

died but a new path has not been established, and therefore no DATA packets are delivered to the sink. Directed Diffusion does not have a QoS pause because all the nodes in the network die at the same time (a sensor's power consumption in transmitting, receiving and idling are the same), thus no redundant routing paths or source nodes are available to use. As a result, Directed Diffusion leads to a short data delivery period. Moreover, compared to ideal receiving, sleeping Directed Diffusion has almost the same increasing slope in throughput except QoS pauses, but SMAC always has lower throughput, because SMAC suffers from more contention and hence more collisions than IEEE 802.11.

4.2 Performance Comparisons of Individual Layer Sleeping

In this section, we examine the performance of sleeping at individual layers. For simplicity, we mention the combination of Directed Diffusion and IEEE 802.11 as DD802, the combination of Directed Diffusion and SMAC as DDSMAC followed by a specific duty cycle, and the combination of sleeping Directed Diffusion and IEEE 802.11 as DDslp802.

Varying Node Density. In this experiment, we vary the node density. Fig. 3 shows the performance rendered by 7 schemes under the deployment of 10 nodes, 20 nodes, 30 nodes, 40 nodes and 50 nodes, within the fixed area. As we can see, without sleeping, DD802 performs the same in both throughput and data delivery ratio, no matter what the node density is. This provides a baseline to judge the performance of all the sleeping schemes. Specifically, DD802 has a throughput of 105 packets on average with 100% data delivery ratio. However, its data delivery period is short in the absence of any sleeping technique.

DDSMAC, on the other hand, shows a diversity of performance under different duty cycles and different node densities. For a given node density, generally, the lower the duty cycle is, the longer alive time every sensor in the network can have, which is beneficial to receiving more packets at the sink node. However, a low duty cycle also leads to severe contention and consequently higher probability of collision. One one hand, collisions may affect the SYNC packet exchange, so that some sensors cannot communicate with each other, and hence have to drop packets. On the other hand, consistent collisions may either lead to packet drop once the packet is retransmitted up to the limit, or incur long packet delay, so that negative reinforcement may be initiated by Directed Diffusion, and then the data delivery path is cut off. Therefore, lower duty cycles always have lower data delivery ratio, but lower duty cycles may not necessarily have high throughput. As shown in Fig. 3, the best duty cycle for DDSMAC in terms of throughput varies according to the node density, which reflects the contention in the network. As the node density increases, the duty cycle with highest throughput increases from 10% for 10 nodes to 40% for 50 nodes. Note that when the node density is very high, more than 40 nodes in our experiment, DDSMAC under all duty cycles performs worse than DD802 in both throughput and data delivery ratio. Therefore, it is not worth sleeping only at the MAC

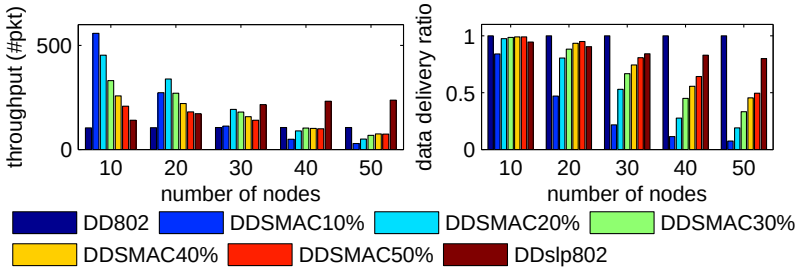

Fig. 3. Performance comparison under varying node density.

layer in this case. For a given duty cycle, both throughput and data delivery ratio decrease as the node density increases. This can be explained by the fact that high node density causes high contention.

On the contrary, DDslp802 has higher throughput as the node density increases. Higher node density implies more routing redundancy. DDSMAC wastes the routing redundancy, but DDslp802 takes advantage of this added redundancy. Specifically, DDSMAC wakes up redundant routing nodes the same as it wakes up reinforced routing nodes. Hence when a reinforced routing path runs out of energy, all the other redundant paths run out of energy as well. The maximum throughput of DDSMAC is upper-bounded by the the total number of packets that can be generated by a single source node. DDslp802, however, saves the energy of those redundant paths by allowing redundant routing nodes to sleep, so that when one path dies, another path can be used to deliver packets. The higher the node density is, the more redundant paths will be available to improve the data delivery period as well as the throughput. However, DDslp802 suffers from QoS pauses. The more often it changes to a new routing path, the more chances it introduces a period of QoS pause. Hence, DDslp802 has worse data delivery ratio as the node density increases. However, the data delivery ratio of DDslp802 decreases much slower than the data delivery ratio of DDSMAC. Fig. 3 also shows that QoS pause impairs the data delivery ratio more than SMAC unreliability at low node density, but it impairs the data delivery ratio less than SMAC unreliability at high node density.

Due to space limitations, the standard deviations of throughput and data delivery ratio are not shown in the figures. However, we observed that DD802 and DDslp802 have very small standard deviations for both throughput and data delivery ratio, while DDSMAC has larger standard deviations as the node density increases or as the duty cycle decreases. Similar results are observed in all the experiments throughout the paper.

Varying Network Scale. In this experiment, we fix the node density, but vary the network scale. 17, 23, 30, 38 and 47 nodes are placed in a 600mX600m, 700mX700m, 800mx800m, 900mX900m and 1000mX1000m area, respectively.

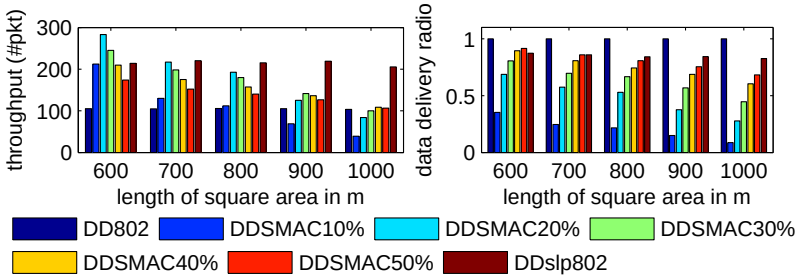

Fig. 4. Performance comparison under varying network scale.

Fig. 4 shows the performance of the 7 schemes under different network scales with the same node density. DD802 performs almost the same as in the experiment of varying node density in a fixed area, with a throughput of 105 packets on average and 100% data delivery ratio. Obviously, neither node density nor network scale influences the performance of DD802.

DDSMAC has an overall decreasing performance as the network scale increases. Since a large network scale implies that source nodes are on average more hops away from the sink node, packets from a source node may experience worse synchronization, longer delay and higher dropping probability on the routing path. Therefore, as the network scale increases, the best duty cycle for DDSMAC in terms of throughput increases from 20% for 600mX600m area to 40% for 1000mX1000m area, due to the fact that larger duty cycles always have better data delivery ratios.

DDslp802, on the other hand, has a steady performance in both throughput and data delivery ratio. As fixed node density guarantees that a sensor has on average a fixed number of neighbors, enlarging the network scale does not increase the routing redundancy. Meanwhile, IEEE 802.11 has reliable data delivery, hence DDslp802 does not suffer from extended multi-hop transmission.

Varying Number of Sources. In this experiment, the number of source nodes varies from 2 to 26. Fig. 5 shows the performance of the 7 schemes. DD802, as usual, has constant performance.

Overall, DDSMAC has an obvious improvement in throughput when the number of source nodes increases from 2 to 5, but the improvement slows down when the number of source nodes continues to increase. As the number of source nodes increases, the sink node can on average reach one of the source nodes in fewer hops. The fewer hops a transmission experiences, the higher data delivery ratio it will have, see Fig. 4. However, as the number of source nodes increases, the probability that the sink node has at least one neighboring source node improves from fast to slowly (more than one 1-hop source node does not help DDSMAC since DDSMAC does not utilize routing redundancy). Accordingly, the improvement in throughput slows down.

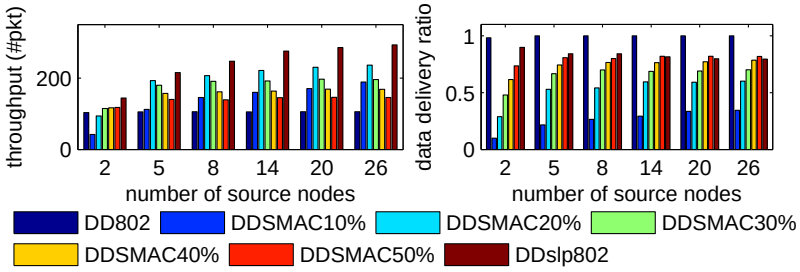

Fig. 5. Performance comparison under varying number of source nodes.

DDslp802 increases in throughput, but decreases in data delivery ratio. This is because source nodes are the second redundancy that sleep Directed Diffusion could use besides redundant routing nodes. Since Directed Diffusion reinforces one path (one source node) at once to deliver data, other redundant source nodes are put to sleep for later use. Hence, once a source node dies, another source node can be reinforced to continue the data delivery. However, the more source nodes to reinforce, the more chances DDslp802 will incur a QoS pause. Consequently, the data delivery ratio decreases slightly as the number of source nodes increases.

Varying Data Rate. In this experiment, the application data rate at each reinforced source node varies from 1 packet per 10s to 8 packets per 10s. Fig. 6 shows the throughput of the 7 schemes scaled by the application data rate and their corresponding data delivery ratio. The throughput of DD802 increases proportionally to the application data rate, while the data delivery ratio is 100%.

In absolute terms, DDSMAC also receives more packets as the application data rate increases, but relatively, the throughput of DDSMAC is decreasing if scaled by the application data rate. The decreasing performance of DDSMAC can also be seen in its data delivery ratio. This is because SMAC cannot provide reliable data delivery, as shown in Section 4.1, due to the contention in the network. Higher application data rates mean more packets to send, and hence more contention and packet loss in a fixed period of time. As the application data rate increases, the best duty cycle in terms of throughput increases from 20% for 1 packet per 10s to 30% for 8 packets per 10s. Larger duty cycles can alleviate contention in the network, as sensors are sleeping for a shorter time.

DDslp802, on the other hand, has throughput proportional to the application data rate and maintains the same data delivery ratio as the application data rate changes. As IEEE 802.11 provides reliable data delivery, the data delivery ratio of DDslp802 drops only because of the QoS pauses introduced by sleeping Directed Diffusion. When the node density and the number of sources are kept the same, the redundancy that DDslp802 can utilize does not change. Hence, DDslp802 reinforces on average the same number of redundant paths, and consequently has the same chances of introducing a QoS pause.

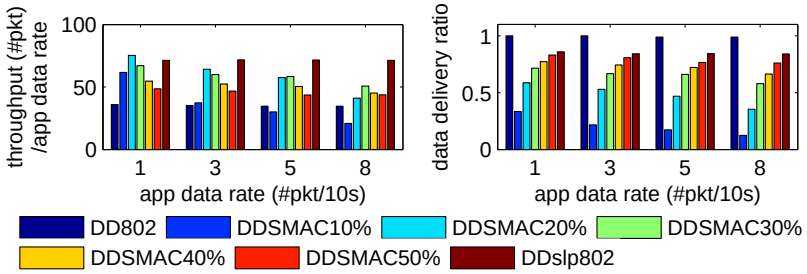

Fig. 6. Performance comparison under varying application data rate.

4.3 Performance of Sleeping at Both Layers

In this experiment, we compare the performance of sleeping schemes at both routing and MAC layers, with and without cross-layer coordination, as described in Section 3.3, as we vary the node density. For simplicity, we mention the combination of sleeping Directed Diffusion and SMAC with its best duty cycle in terms of throughput under DDSMAC as DDslpSMAC, followed by -NC or -C for the case of without coordination or with coordination, respectively.

Fig. 7 shows the throughput and the data delivery ratio of 5 schemes. As we can see, DDslpSMAC-NC has very similar performance in both throughput and data delivery ratio as DDSMAC even though sensors can now sleep at both layers. Since no coordination is implemented between the routing layer and the MAC layer, sleeping Directed Diffusion does not have higher priority than SMAC to schedule the sensors. When a sensor is turned off by sleeping Directed Diffusion, it will still be woken up periodically by SMAC according to its duty cycle. Hence, sensors are mostly following SMAC to sleep and wake up.

On the other hand, with certain coordination, DDslpSMAC-C significantly improves the throughput compared with DDSMAC, since it not only reduces idle listening during data delivery but also utilizes network redundancy. When the node density is not very high, DDslpSMAC-C performs the best in terms of throughput among all the sleeping schemes. However, when the node density increases, DDslpSMAC-C cannot achieve as high throughput as DDslp802, since the severe contention at the MAC layer greatly impairs the most vulnerable but most important data delivery at the routing layer - unicasting positive reinforcement packets and unicasting high rate DATA packets. The loss of positive reinforcement packets delays the path establishment, while the loss of DATA packets impairs the throughput directly. Hence, DDslpSMAC-C cannot further improve the throughput compared with DDslp802, although sensors sleep at both layers. DDslpSMAC has a data delivery ratio lower than the data delivery ratio of either DDSMAC or DDslp802, because both QoS pauses and SMAC unreliability are degrading the performance. Meanwhile, DDslpSMAC has similar standard deviations with DDSMAC in either throughput or data delivery ratio, while DDslp802 has smaller standard deviations (not shown in the figure).

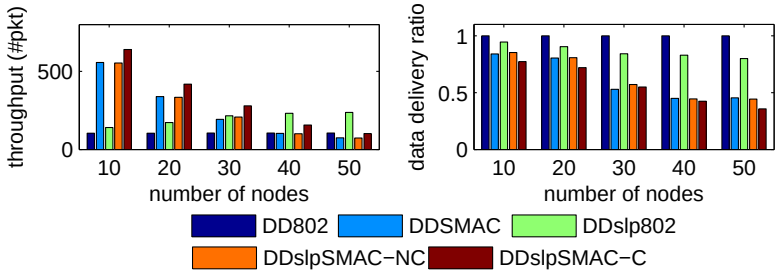

Fig. 7. Performance of multi-layer sleeping with/without cross-layer coordination.

These preliminary results show the benefit of employing cross-layer coordination between the routing and MAC layers to achieve higher throughput under low network density. Hence, a smart decision can be made by the power management of wireless sensor networks to dynamically choose the best sleeping scheme (single layer sleeping or multi-layer sleeping with coordination) as the network density changes over time. We expect in our future work that sleeping at both routing and MAC layers with cross-layer coordination can also outperform single layer sleeping schemes when varying network scale, the number of source nodes and the application data rate, as long as the contention in the network is low. Moreover, we believe the performance of DDslpSMAC-C can be further improved and may outperform single layer sleeping schemes all the time as more sophisticated cross-layer coordination is employed.

5 Conclusions

In this paper, we analyze sleeping schemes conducted by routing protocols and MAC protocols individually and simultaneously, for wireless sensor networks, in order to determine the best method for sleeping under different network and application scenarios. While conclusions are made by simulating networks that run routing protocols with/without sleeping (Directed Diffusion and our newly proposed sleeping Directed Diffusion) and MAC protocols with/without sleeping (IEEE 802.11 and SMAC), the conclusions can also be applied to other routing protocols that turn off sensors when they are not involved in the data delivery, and other MAC protocols that use duty cycles to save energy. In general, our results show that routing layer sleeping is more suitable for networks with high redundancy or high contention, while MAC layer sleeping is more sensitive to contention, and hence is a good choice for light traffic applications under small scale networks. Furthermore, we show that cross-layer coordination can significantly improve the network throughput under low contention scenarios, when routing layer sleeping and MAC layer sleeping are employed simultaneously. Therefore, a smart decision can be made by a power manager to dynamically switch to a sleeping scheme at the routing layer or the MAC layer or both layers

with cross-layer coordination, as the network conditions or application requirements change over time. Moreover, more sophisticated cross-layer coordination has the potential to further improve the network throughput and outperform single layer sleeping schemes in all cases.

Acknowledgments. This work was supported in part by the National Science Foundation under grant # CNS-0448046 and in part by a Young Investigator grant from the Office of Naval Research, # N00014-05-1-0626.

References

1. Zoghi, M.R., Kahaei, M.H.: Sensor Selection for Target Tracking in WSN Using Modified INS Algorithm. In: 3rd Internatioanl Conference on Information and Communication Technologies: From Theory to Applications, pp. 1-6. 2008
2. Xu, Y, Heidemann, J., Estrin, D.: Geography-informed Energy Conservation for Ad Hoc Routing. In: 7th Annual International Conference on Mobile Computing and Networking, pp. 70-84. 2001
3. Chen, B., Jamieson, K., Balakrishnan, H., Morris, R.: Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wirleess Networks. Wireless Networks, vol. 8, issue 5, pp. 481-494. Kluwer Academic Publishers 2002
4. Zheng, R., Kravets, R.: On-demand Power Management for Ad Hoc Networks. In: 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, vol.1, pp. 481-491. 2003
5. Wang, H., Wang, W., Peng, D., Sharif, H.: A Route-oriented Sleep Approach in Wireless Sensor Network. In: 10th IEEE Singapore International Conference on Communication systems, pp. 1-5. 2006
6. Ye, W., Heidemann, J., Estrin, D.: Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks. IEEE/ACM Trans. on Networking, vol. 12, issue 3, pp. 493-506. 2004
7. Polastre, J., Hill, J., Culler, D.: Versatile Low Power Media Access for Wireless Sensor Networks. In: 2nd International Conference on Embedded Networked Sensor Systems, pp. 95-107. 2004
8. Kulik, J., Rabiner, W., Balakrishnan, H.: Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In: 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, pp. 174-185. 1999
9. Intanagonwiwat, C. Govindan, R. Estrin, D. Heidemann, J.: Directed Diffusion for Wireless Sensor Networking. IEEE/ACM Trans. on Networking, vol. 11, issue 1, pp. 2-16. 2003
10. Chang, J. Tassiulas, L.: Maximum Lifetime Routing in Wireless Sensor Networks. IEEE/ACM Trans. on Networking, vol.12, issue 4, pp. 609-619. 2004
11. Perillo, M., Heinzelman, W.: DAPR: A Protocol for Wireless Sensor Networks Utilizing an Application-based Routing Cost. In: Wireless Communications and Networking Conference, vol. 3, pp. 1540-1545. 2004
12. Luo, H., Luo, J., Liu, Y., Das, S.K.: Adaptive Data Fusion for Energy Efficient Routing in Wireless Sensor Networks. IEEE Trans. on Computers, vol. 55, issue 10, pp. 1286-1299. 2006
13. http://circuit.ucsd.edu/ curts/courses/ECE284_F05/lectures /ECE284_F05_L07_EnergyEfficiency_2pp.pdf
14. http://www.isi.edu/nsnam/ns/