Low Power Coordination in Wireless Ad-hoc Networks

F. Koushanfar, A. Davare, D.T. Nguyen
EECS Dept. UC Berkeley
{farinaz,davare,nguyendt}@eecs.berkeley.edu

M. Potkonjak
CS Dept.
UC Los Angeles
miodrag@cs.ucla.edu

A. Sangiovanni-Vincentelli
EECS Dept. UC Berkeley
alberto@eecs.berkeley.edu

ABSTRACT

Distributed wireless ad-hoc networks (DWANs) pose numerous technical challenges. Among them, two are widely considered as crucial: autonomous localized operation and minimization of energy consumption. We address the fundamental problem of how to maximize life-time of the network by using only local information while preserving network connectivity. We start by introducing the Care-Free Sleep (CS) Theorem that provides provably optimal necessary and sufficient conditions for a node to turn off its radio while ensuring that global connectivity is not affected.

The CS theorem is the basis for an efficient localized algorithm that decides which node will turn its radio off, and for how long. The effectiveness of the approach is demonstrated using numerous simulations of the performance of the algorithm over a wide range of network parameters.

Categories and Subject Descriptors

C.2.2 [**COMPUTER-COMMUNICATION NETWORKS**]: Network Protocols; C.4 [**PERFORMANCE OF SYSTEMS**]: [Reliability, availability, and serviceability]

General Terms

Algorithms, Design, Performance

Keywords

Wireless ad-hoc network, low-power, coordination

1. INTRODUCTION

Distributed wireless Ad-hoc Networks (DWANs) are distributed embedded systems consisting of a large number of nodes each equipped with computational, storage, communication, and in some cases, sensing and actuating subsystems. While DWANs open up numerous, high impact research and economic opportunities, they pose several new challenging technical problems. There is a wide consensus that among these problems, two are of dominating importance: (i) low energy design and operation and (ii) autonomous localized operation and decision making. Recent studies have shown that one of the most effective methods for energy conservation in DWANs is to reduce the idle energy consumption of the nodes. This is achieved by putting a large percentage of nodes into sleep state by turning off the nodes' radios [3, 7]. At the same time, one should ensure that the sleeping nodes are not required for addressing the current network and users needs.

Although there have been a number of efforts to determine the conditions for a node to enter sleep state using only locally available information while preserving the overall connectivity of the network, only heuristic answers have been presented [17, 3, 1]. We refer to this problem as the *sleep coordination problem.* The sleep coordination problem is interesting and challenging from several view points:

- **Complexity of the Problem.** The nodes that stay awake to preserve the connectivity of the network form a connected dominating set on the network graph. Finding the minimum connected dominating set can be proven to be NP-complete. Therefore, even in cases where we do have the complete graph information about the whole network, finding the optimal solution in polynomial time is unlikely. Furthermore, setting the proper sleep times to the nodes to maximize the overall network's lifetime, adds a new dimension to the NP-complete minimum connected dominating set problem.
- **Scope of the Problem.** For a sleep coordination procedure, making a globally sound decision using only local information is a challenging task. Changing the status of even one node can potentially impact any node in the network in terms of its connectivity and energy consumption.
- **Guaranteed Connectivity.** There is a need to determine under which conditions, a particular node can sleep, while still guaranteeing that the network is connected.
- **Protocol Design.** The autonomous operation of the nodes in DWANs has several advantages including fault tolerance, fast response to changes, and non-preplanned network structure. However, interaction and collaboration between the nodes and existence of shared resources, dictates a need for a protocol that can handle concurrency and synchronization of the autonomous ad-hoc node decisions.

The power saving coordination strategy introduced here attempts to address these challenges. We start by introducing the Care-Free Sleep (CS) theorem that establishes provably optimal necessary and sufficient conditions for a given node to enter sleep state without disconnecting the network.

We use the CS theorem as a starting point to develop algorithms for determining which nodes should be put into sleep and for how long they should stay in that state. Also, we show how the algorithms could be localized to a user specified level. The algorithm takes into account, simultaneously, both the amount of energy at each relevant node and the minimization of the number of of hop-communication between any two nodes that are awake. Finally, we introduce a localized synchronization protocol that provably avoids deadlocks, guaranteeing that the connectivity of the network is maintained.

To introduce the core problem of determining when a node can go to sleep such that it meets connectivity requirements, we present a small and illustrative example. Figures 1.(a) and 1.(b) show a part of a network. It is assumed that nodes M, L, J, I, H, F, G, D, and N are connected to other nodes in the network. A square indicates an awake node and a triangle indicates a sleeping node. The edge between two nodes indicates that they can communicate if both nodes are awake. We assume that each node knows only about its neighbors' existence and the neighbors' awake/sleep status. In this example, we are trying to determine if node A can go into sleep state. Assuming that communication has a dominating energy cost [10], we want to contact as few nodes as possible, which limits us to use only local information. The key observation is that this task can be done locally: all that is needed is to check if there is a path that visits all awake neighbors of node A. In Figure 1.(a), one such path is highlighted. In Figure 1.(b), there are no such paths, thus node A should not go into sleep state. The reason is that if node A goes to sleep, nodes C and B will become disconnected from the rest of the network. Note that the only difference between the two networks is that in Figure 1.(a) node N is awake, and Figure 1.(b) it is sleeping. In the rest of the paper, we demonstrate how this observation can form the basis for an efficient method to optimize the life-time of a network.

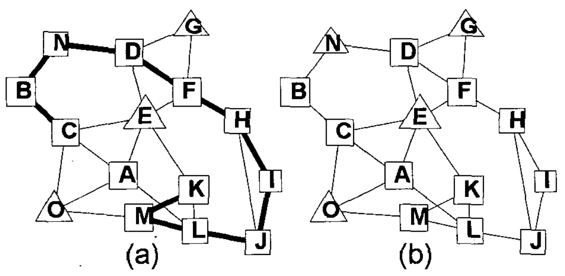

Figure 1: Example of a low-power coordination

2. RELATED WORK

Energy awareness is an active area of research for battery-operated wireless ad-hoc networks. Energy awareness can be adopted in all layers of the network [10], ranging from physical layer [14, 9], link layer [13], and MAC layer [18, 8], to network and application layers [8, 17, 1].

The first important step in selecting an energy efficient protocol is finding the underlying communication energy consumption model of the components of the network. There is a wide consensus that the energy used by the communicating devices (radios) is much higher than the energy used in computations [10, 14], even when the radios are in an idle mode [16, 7]. In this work, we use the radio energy consumption model by Kasten [7] from the various studies on the Digitan 2Mb/s 802.11 wireless LAN.

Power aware routing in ad-hoc networks is another active area of research [11, 2]. Using a minimum connected dominating set for routing messages in ad hoc networks have been proposed and a number of heuristics have been developed [4]. More recently, a number of power minimization methods using a sleeping strategy were proposed for wireless ad-hoc networks. GAF [17] is a conservative energy conservation scheme that superimposes a virtual grid proportional to the communication radius of the nodes onto the network. Since nodes in one grid are equal from the routing perspective, the redundant nodes within a grid can be turned-off. SPAN [3] is a distributed randomized algorithm that attempts to preserve connectivity in wireless networks, but does not provide a proof of necessary and sufficient conditions for guaranteeing this preservation of connectivity. ASCENT [1] proposes adaptive self-configuration algorithms for wireless networks that enables online tuning of the system parameters, such as sleep time of the nodes, to extend the lifetime of the overall system. STEM [12] suggests another power saving strategy that does not try to preserve the connectivity of the network. STEM works by putting an increasing number of nodes into sleep state, and then encountering the latency to set up a multi-hop path. Nodes in STEM must have an extra low power radio called a *paging channel* that does not go into sleep state and constantly monitors the network to wake the node up in the cases of an interesting event.

3. PRELIMINARIES

We assume an ad-hoc wireless network where there is no fixed infrastructure or centralized wireless control. Due to the limited energy sources, a node only communicates directly to other nodes within its short-range distance. The longer distance communications are performed using other nodes as intermediate relays. Although we have used this model in our simulations, our algorithm is generic in that it can work with any communication model as long as the network is connected and there is least one path between every two nodes in the network. The network is assumed to be fully connected initially. The communication between the nodes is in the form of broadcasting. Also, a node that is in the awake state periodically broadcasts a POLLING messages that contains information about that node which includes its ID, current token group (described in Subsection 5.3), its current energy level and location along with a list of its current neighbors. Our algorithm is operating above the link layer and medium access (MAC) layers of the network, so it can take advantage of the power saving strategies proposed for those levels.

A node can be in one of the two following states. (1) A node that has its radio on is in *awake* state. A node in awake state can either listen to the traffic in the network (*idle*) or transmit and/or receive messages (*active*). (2) A node is assumed to be in *sleep* state, if its radio is off and hence it is disconnected from the communication topology graph. A node can switch its state from time to time, and the energy usage of a node depends on its current state. We assume the energy model in [7] where a node's energy consumption in *sleep* state is much lower than the energy consumed in other situations. We further distinguish between the energy consumed for listening (or idle), transmission and reception.

4. PROBLEM FORMULATION

In this section, we present informal and formal definitions of the overall addressed problem - sleep coordination. At the heart of the problem is a question that is the focus of this section: when a given node can go into sleep state in such a way that the remainder of the awake nodes form a connected network and that each node in the sleep state has at least one neighbor awake. The rationale behind the first condition of this formulation is to allow all awake nodes to communicate. The rationale of the second condition is twofold. First, when a node wakes up, it can immediately access information that was sent to it through one of the awake neighbors. Second, if a node has to wake-up, the node can immediately communicate to any awake node. We conclude this section by stating and proving the Care-Free Sleep Theorem that establishes necessary and sufficient conditions for a node to go to sleep while maintaining the connectivity of the network. Formally, the overall sleep coordination problem can be stated in the following way.

Assumptions:

- An ad-hoc network with N nodes which form a connected graph.
- Each node i has an amount of energy U_i initially.
- A node spends the amount of energy (per unit of time), shown by U_s, U_d, U_t, U_r , while sleeping, idle, transmitting and receiving respectively. *(Note that U_s is significantly lower than U_d, U_t, U_r.)*

INSTANCE: *Graph $G = (V, E)$, function $U(i)$, for all i elements of V, and an integer T.*

QUESTION: *Is there a mapping $F(i,t)$ to binary domain s.t. for each t all nodes i s.t. $F(i,t) = 1$ have an edge between them, and that for all nodes j that have $F(j,t) = 0$, there is an edge to a node k that has $F(k,t) = 1$. In addition, for all nodes in the graph $\sum_i(F(i,t) \leq U)$, and $t \geq T$.*

We have proven that the sleep coordination problem is an NP-complete problem by transforming the connected dominating set problem [6] into a special case of the sleep coordination problem. At the core of the sleep coordination problem is a Care-Free Sleep subproblem.

Care-Free Sleep Theorem: *A node can go into sleep if and only if there exists a path that visits all of its awake neighbors and all of its sleeping neighbors have at least one other awake neighbor.*

Proof. If:

Suppose the conditions are satisfied. Note that the existence of a path that visits all the neighboring nodes implies a path between any pair of neighbors. Let us denote two arbitrary nodes in the network by S and D. Assume that there is a path P between S and D and that the path P is through node A. Obviously, path P is incoming to node A from one of its neighbors and leaving through another neighbor. We denote this neighbors as N_i and N_o. After node A goes to sleep, by the condition of the theorem there is still a path P_{io} from N_i to N_O that does not contain A. Therefore, if we replace the original subpath $N_i \rightarrow A \rightarrow N_o$ by a subpath P_{io}, the later subpath will form a path from S to D with the reminder of the path P $(P \setminus ((N_i \rightarrow A \rightarrow N_o))$. Since each of the sleeping neighbors of node A has another neighbor awake, the second condition will be also satisfied.

Only if:

Suppose the conditions are not satisfied. Therefore, there is a pair of neighbors N_i and N_o that do not have a path between them after the node A goes to sleep or there is a neighbor node N_s in sleep state that has A as its only awake neighbor. The first condition implies that N_i or N_o (or both) will be isolated (not connected to the rest of the network) after node A goes to sleep. The second condition implies that node N_s will not have an awake neighbor. Therefore, node A must stay awake.

5. LOW ENERGY PROTOCOL

The approach to change the state of nodes to the sleep and rotating the awake/sleep nodes for a connected power-saving network are described in this section. Initially, nodes start in the awake state, idle mode. The idle mode changes as the nodes poll the network to discover their immediate neighborhood by exchanging routing and control messages. Also, in case an unusual event occurs in the environment, it generates traffic to route the event to interested nodes that require the information.

5.1 Node Transition to Sleep State

To enable multiple threads to control the state transition algorithm, we introduce the notion of a *token*. A *token* defines the current computing node (*CCN*) that has the control of the coordination procedure. Since the procedure has a localized scope and a distributed nature, there can be multiple tokens operating in different places in the network. The tokens are assigned to nodes and handshaking between the tokens is done using the protocol presented in Subsection 5.3. Pseudocode of the localized procedure at a *CCN* for putting a node to sleep is summarized in Figure 2. A *CCN* makes an evaluation of itself and its awake token group members in the neighborhood to decide which one of them is the best choice for transition to the sleep state (Step 1). Before making any decisions, the information about the routes in the k-hop neighborhood of the *CCN* should be updated (Step 3). The updating is accomplished with series of local exchanges between neighbor nodes. The procedure reserves a variable named *BEST-CANDIDATE* consisting of a node's ID and its critical energy at the *CCN* and assigns the value zero to critical energy and nil to ID (Step 4). While the computation is still conducted at *CCN*, the procedure evaluates node *CCN* and each of its neighbors in the group to determine the best candidate for the sleep state (Steps 5-17). Note that all the computation at the *CCN* requires only information from the k-hop routing table gathered in Step 3. The evaluation process for a *NODE(i)* starts by computing the shortest path between each pair of its neighbors using the standard Floyd-Warshall algorithm, without using *NODE(i)* on the path (STEP 7). If there are a pair of neighbor nodes that do not have a path between them, *NODE(i)* cannot go to sleep state (Steps 8-9). Otherwise, the procedure selects the pair of neighbors to the *NODE(i)* with the longest path between them and saves the path information in *MAX-SHORTEST(i)* (Steps 10-12). Next, the node with the minimum energy on the shortest path is selected as the critical node and its information is saved in *CRITICAL-U(i)* (Step 13). This node is critical since alteration of its state can cause the alternative path to become

```
Procedure Localized-Sleep-Coordination
1. At each CCN node with a token do
2. {
3.   Update the k-hop routing table of (CCN) for
     non-locked nodes and lock the used nodes;
4.   BEST-CANDIDATE ← (0,nil);
5.   for NODE(i) = CCN to (CCN-group-neighbors) do
6.   {
7.     Calculate the shortest path between each pair of
       NODE(i) neighbors in k-hop neighborhood;
8.     If (no path for a pair of neighbors to NODE(i));
9.       NODE(i) cannot go to sleep state;
10.    else
11.    {
12.      MAX-SHORTEST(i) ← max shortest
         path between pairs of neighbors to NODE(i);
13.      CRITICAL-E(i) ← node with min
         energy on MAX-SHORTEST(i);
14.    }
15.    if (CRITICAL-U(i) > BEST-CANDIDATE)
16.      BEST-CANDIDATE ← (CRITICAL-U(i),i);
17. }
18. Select the BEST-CANDIDATE node for sleep;
19. Select an awake neighbor to CCN that has
    not received the token for the longest time;
20. Broadcast ID of sleeping node, remove locks
    and pass the token to next token node;
21.}
```

Figure 2: Pseudocode for the localized coordination strategy to select the nodes to sleep.

disconnected. Since the goal is to maximize the network lifetime, if the alternative path to a sleeping node is critical in energy, that node is not a good candidate for sleep state. Therefore, *BEST-CANDIDATE* always stores the node with the maximum *CRITICAL-U(i)* (Step 18). After that, the *CCN* has to decide the next token node from among its neighbors. The procedure selects an awake neighbor in its group that has not received the token for the longest time (Step 19). The *CCN* node then broadcasts its decision about the sleep transition, frees the nodes in its k-hop neighborhood and passes the control of the procedure to the next token node (Step 20).

5.2 Sleep to Idle Transition

A node that transitions to the sleep state, sets a self-timer and remains in this state for T_s time units. After the interval T_s, the node wakes up and sends POLLING messages to the nodes in its proximity. The duration of sleep time T_s, is important for at least three reasons. First, the longer the T_s, the more the power savings. Note that the transition from sleep state to awake state could be energy intensive. Second, there is an uncertainty in estimating the rate of energy consumption of the nodes on the alternative paths during the time T_s, due to the non-linearity of battery discharging and also the random traffic patterns in the network. Third, the shorter the length of T_s, the more flexible the network is to network dynamics (e.g. handling node mobility). We determine T_s as a function of (i) energy of the alternative paths to the node, (ii) the expected traffic energy consumption, and (iii) the length of the alternative paths to one node. We adopt the results of the experiments for tuning the parameter with respect to different metrics in the network. In our experiments, T_s is tuned with respect to density, the overall lifetime of the network and the percentage of tokens used.

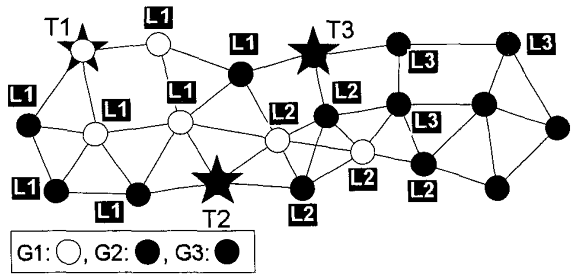

Figure 3: An example of token synchronization between token groups: $T1$, $T2$ and $T3$ denote the *CCN* nodes in corresponding groups. $L1$, $L2$ and $L3$ are the nodes locked by $T1$, $T2$ and $T3$ respectively.

5.3 Synchronization Protocols

Tokens are assigned to the nodes during the network setup. They are updated via the POLLING mechanism. Initially, each node generates randomly a token with a user specified probability. The generator node's ID is used as the name of its group to resolve conflicts. Next, the token node invites the nodes in its neighborhood to join its group, unless they have already joined another token group. The token invitation continues to propagate further until there is no node adjacent to the nodes in the group without a token. If a token group ends up as a single node group, it randomly merges itself into one of its adjacent token groups. A node stays in a token group as long as it has at least one neighbor that belongs to the same token group. Otherwise, the node sends a nil in its POLLING message and searches for a token group in its proximity to join.

At each point of time, each token makes a localized decision for the nodes in its group using the coordination procedure shown in Figure 2). There might be a conflict between the tokens as there might be shared nodes that are within the k-hop neighborhood of two *CCN*'s from two different tokens. In order to resolve the conflict, the procedure adapts a semaphore-based [5, 15] synchronization and resource allocation strategy, where the shared resources are only assigned to one token group at a time. During the k-hop neighborhood discovery (Step 3), the node only gathers information from unlocked nodes, which are the ones that are not in use by other tokens. Each token locks the nodes that are not locked in its k-hop neighborhood before making the decision as to which node to put to sleep in Steps 5-20. Note that there is a need for a priority scheme among the tokens: we give the priority to the token group with the smaller ID. A small example for token synchronization and locking mechanisms is shown in Figure 3. The members of the groups are shown using different fillings. The nodes denoted by $T1$, $T2$ and $T3$ are *CCN*'s for $G1$, $G2$ and $G3$ respectively. In order for the example to be small, the *CCN* nodes consider only their 2-hop neighborhood. The black labels $L1$, $L2$ and $L3$ show the nodes that are currently locked by $T1$, $T2$ and $T3$ respectively. As can be seen here, tokens can lock the nodes in other groups and use them in their alternative paths, although a token cannot assign state to a node that belong to another token group.

6. EXPERIMENTAL RESULTS

In this Section, we present our experimental results for evaluation of our localized power saving sleeping strategy. The simulation environment consists of more than 3,000 lines of simulation code in C++. Since there are a number of parameters involved in this study, we gathered a large database with more than 10,000 random instances to statistically study the performance of our algorithm. We study the lifetime of the network and measure how long the network is operational before it becomes partitioned into several disconnected parts. We show this measure as a percentage increase over the lifetime of the network without the sleeping strategy. Note that previous research in this area such as SPAN [3] and GAF [17] have used a different metric for measuring the lifetime of the network, i.e. the fraction of nodes with non-zero energy. We have considered a variety of k-hop neighborhood scopes for each token in Step 3 of Figure 2, although in the studies shown here, we use only a 3-hop neighborhood.

The network is generated by randomly placing the nodes in a 1×1 unit square. The communication range of the nodes is specified by the user. Since changing the ranges and the number of points in a fixed area both generate scaled versions of the same density, we fixed the ranges to 0.13 units in these experiments and only alter the number of nodes to change the densities. Initially, every node has an energy of 500J. We assume a traffic model that consists of a continuous bit rate (CBR) traffic generator that spreads the traffic randomly among 15% of the nodes. The sleeping time T_s is defined as a function of the percentage of the expected energy depletion from the nodes on the alternative paths to the node under considerations. The number of tokens is also calculated as a percentage of the number of nodes in the network. For the energy consumption model, we adopted the numbers from the study in [7], from results obtained from the Digitan 2Mb/s 802.11 wireless LAN. The power consumption numbers according to this model are: transmission (1.9W), reception (1.5W), listening (0.75W) and sleep (0.025W).

From our experiments, the sleeping algorithm yields a significant power savings, especially for cases with high node densities. This corresponds to natural intuition about the problem: In higher densities, more nodes can back-up each other's functionality and therefore, more nodes can go to sleep and make the lifetime of the network longer.

Figure 4 shows the number of alive nodes, as the simulation time progresses. The original network is deployed in a 1×1 area using 300 nodes, each with the communications range 0.13. Without a sleeping schedule, the expected lifetime of a node, and thus the network, with traffic is less than 666s. As shown on the Figure, the lifespan of some nodes have increased by a factor of three or four. Table 1 illustrates the average performance of our studies over a range of different node densities, various numbers of tokens and, different sleeping times. In Table 1, the first column shows the number of nodes, the second column shows the number of tokens used by the algorithm, the third column shows the sleeping time and the last column shows the lifetime of the network, averaged over at least 200 instances for each of the settings. As can be seen in the table, although the lifetime increases with increasing network density, there is a wide range of variation over the different parameters involved. Also, the network lifetime increase between 50%-100% for medium to average densities (around 200 nodes in the area) and more than 200% for slightly larger densities (around 300 nodes or more). To analyze these relationships and dependencies, we have used 3-D plots of the lifetime of the network based on the three parameters involved, i.e. number of tokens, sleeping time and the density of the network. Figures 5, 6 and 7 demonstrate these relationships. As we can see in Figures, almost all our experiments are confirming the result that our algorithm is capable of putting more nodes to sleep as the density of the nodes increases to save more power. This is in contrast to SPAN [3] algorithm that has lower power savings comparable to ours in the lower densities but does not scale well with the increase in the density. Since our stopping criteria is when the network is partitioned, existence of a lot of sleep nodes can make the network very dependent on a few awake nodes. Once the power of those few nodes depletes, the network will disconnect, even though sleep nodes can make a connection between the partitions when they wake up. For this reason, in Figure 5, although sleeping times more than 50% can potentially save more power, they are not practical since they might cause temporary partitions in the network. The best results for power saving are for sleeping times around 25%.

Figure 6 shows the variation of the lifetime when the density and the percentage of tokens changes. The network saves more power for smaller percentage of tokens. This is because when there are for example 50% tokens in the network, the expectation of number of nodes in each group becomes 2. At each point of time, at least one member of each token group must stay awake, i.e., around 50% of the nodes. However, there is a minimum number of tokens re-

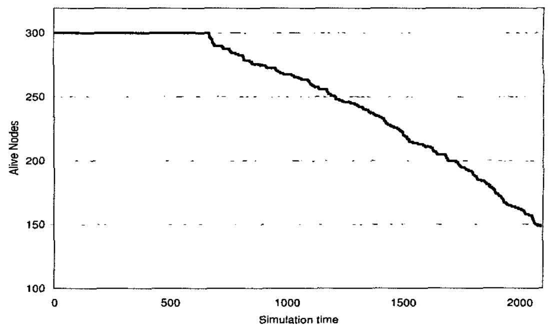

Figure 4: The number of alive nodes as a function of the simulation time.

# of Nodes	% of tokens	T_s (%)	lifetime increase (%)
100	15	20	1.32
200	15	30	51.24
400	15	40	373.46
100	25	20	0.29
200	25	30	82.11
400	25	40	462.67
100	30	20	0.13
200	30	30	73.63
400	30	40	341.30

Table 1: Experimental results, showing the relationship between number of nodes, number of tokens, sleeping time (T_s) and the lifetime of the network.

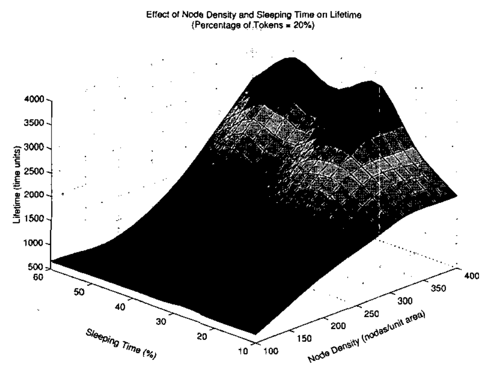

Figure 5: Effect of the network density and sleeping time of the nodes on the lifetime of the network.

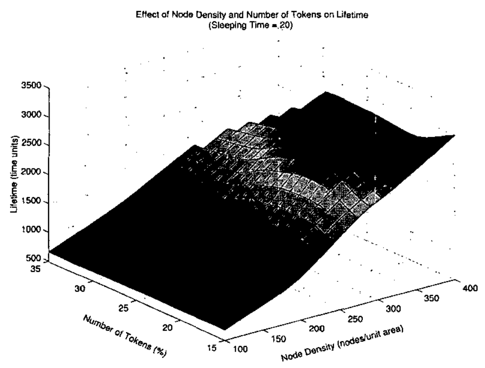

Figure 6: Effect of network density and percentage of tokens on the lifetime of the network.

quired for each density to ensure that nodes can schedule their activities on time by receiving enough tokens. So, a good engineering practice is to tune the percentage of tokens to different node densities. Figure 7 illustrates the relationships between the sleeping time and percentage of tokens with the lifetime of the network. Although there is a mode and maxima for the variations in the graph, the mode is not monotone and an efficient system design study requires adaptively tuning the parameters involved with respect to the different variables involved.

7. CONCLUSION

We have developed a new localized distributed approach for power minimization in wireless systems. By leveraging the Care-Free Sleep theorem that states provably optimal necessary and sufficient conditions for nodes to be placed in sleep state, we have developed algorithms and synchronization protocols to efficiently put a large number of nodes in the sleep state while preserving the connectivity of the network. The effectiveness of the approach is demonstrated using extensive simulations.

8. REFERENCES

[1] A. Cerpa and D. Estrin. Ascent: Adaptive self-configuring sensor networks topologies. In *IEEE Infocom*, 2002.

[2] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networking. In *IEEE Infocom*, 2000.

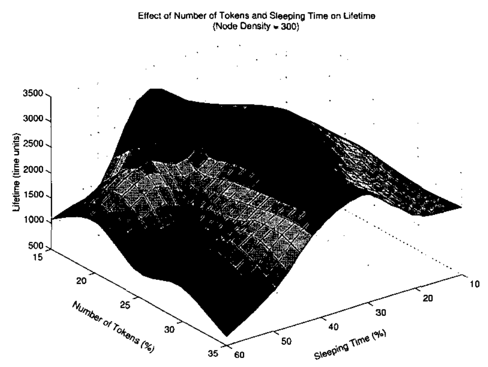

Figure 7: Effect of sleeping time and percentage of tokens on the lifetime of the network.

[3] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Wireless Networks*, 8(5):481–494, 2002.

[4] B. Das and V. Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *ICC (1)*, pages 376–380, 1997.

[5] E. Dijkstra. The structure of the 't.h.e.' multiprogramming system. *Communications of the ACM*, 18(8):453–457, 1968.

[6] M. Garey and D. Johnson. *Computer and Intractability: A Guide to the theory of NP-Completeness.* W. H. Freeman & Co., New York, NY, 1979.

[7] O. Kasten. Measurements of energy consumption for digitan 2 mbps wireless lan module (ieee 802.11/ 2mbps). http://www.inf.ethz.ch/ kasten/research, 2001.

[8] G. J. Pottie and W. J. Kaiser. Embedding the internet: wireless integrated network sensors. *Communications of the ACM*, 43(5):51–58, 2000.

[9] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, and T. Tuan. Picoradios for wireless sensor networks: The next challenge in ultra-low-power design. In *ISSCC*, 2002.

[10] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava. Energy aware wireless microsensor networks. *IEEE Signal Processing*, March 2002.

[11] E. Royer and C. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. In *IEEE Personal Communications*, 1999.

[12] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Topology management for sensor networks. In *ACM MOBIHOC*, pages 135–145, 2002.

[13] E. Shih, S.-H. Cho, B. Calhoun, and A. Chandrakasan. Energy-efficient link layer for wireless microsensor networks. In *Proceedings of the Workshop on VLSI*, 2001.

[14] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In *ACM Mobicom*, pages 272–287, 2001.

[15] A. Silberschatz, P. Galvin, and G. Gagne. Operating system concepts : Windows xp update, 2003.

[16] M. Stemm and R. H. Katz. Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Trans. on Comm*, E80-B(8):1125–31, 1997.

[17] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *ACM MOBICOM*, pages 70–84, 2001.

[18] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *IEEE Infocom*, 2002.