LISP: A Link-Indexed Statistical Traffic Prediction Approach to Improving IEEE 802.11 PSM

Chunyu Hu
Email: chunyuhu@uiuc.edu
Department of Electrical and Computer Engineering
Jennifer Hou
Email: jhou@cs.uiuc.edu
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

Power management is an important technique to prolong the lifetime of battery-powered wireless ad hoc networks. The fact that the energy consumed in the idle state dominates the total power consumed on wireless network interfaces motivates use of protocols that put the radio into the sleep mode in the lack of communications activities. IEEE 802.11 PSM is a representative of such protocols. However, the performance of IEEE 802.11 is significantly degraded under PSM because of the wake-up latency thus introduced.

In this paper, we propose a novel and complementary mechanism, named by *Link-Indexed Statistical traffic Predictor (LISP)* to improve IEEE 802.11 PSM. Essentially LISP employs a simple, light-weight traffic prediction method and enables each node to seek the inherent correlation between ATIM_ACKs and incoming traffic. Once such a correlation is identified, a node en route stays awake in the beacon interval (BI) in which a packet is anticipated to arrive, thus bridging a "freeway" for the packet to rapidly traverse the route. In this manner, a packet can travel from the source to the destination within one BI ideally. Meanwhile, the number of duty cycles is minimized and more energy is conserved. LISP differs from previous work [12], [2], [14] in that we do not trade energy for better end-to-end performance at light to moderate traffic loads, i.e., we reduce the end-to-end delay without consuming more power. Instead more energy is saved because of the reduction in duty cycles. We conduct analytical and simulation studies. The impact of various factors, including traffic load, number of hops, ATIM window size and packet sizes are investigated using a tandem topology. In a general scenario simulated, LISP demonstrates the improvement over PSM in the end-to-end delay by 65-75%, and an increasement of the energy efficiency over IEEE 802.11 with and without PSM by 6% and 178%, respectively.

I. Introduction

Energy is an indispensable but limited resource in battery-powered wireless networks. As wireless communication consumes significant energy of mobile computing devices, how to enable wireless communications to be energy efficient has recently attracted much research attention. This issue is particularly important in wireless ad hoc networks, because wireless devices in such networks serve not only as sources or destinations, but also as relay nodes.

There exist several design choices to tackle the power efficiency issue in wireless ad hoc networks, ranging from designing energy efficient wireless network interfaces to reforming the network protocol stack at various layers. The former saves power by using low power consuming components, and provides upper layers an interface to manage the operating mode of the device. The latter leverages the interface provided by the former and tunes the operations of higher protocol layers to be power-aware. For example, the MAC layer can control the operational mode of the device [6] and/or exercise various power management strategies. (We will provide a comprehensive overview in Section II.) Topology control – determination of the adequate transmission power of each node so as to maintain network connectivity while consuming the minimum possible power [2], [5], [12] – is another (orthogonal) strategy for energy efficiency. In the network layer, several energy efficient routing protocols have been proposed, e.g., minimum energy routing [7] and power aware routing [9], just to name a few. In this paper, we focus on the power saving issue in the MAC layer.

Most of the power saving mechanisms in the MAC layer take into account of energy consumption behaviors of network interfaces. The energy consumption of network interfaces can be typically classified into five modes: transmit, receive, idle, sleep and turn off. In the idle state, a network interface can transmit or receive packets. When switched to the sleep mode, it cannot perform any wireless communications. The energy consumed by the first three modes, therefore, are significantly larger than that in the sleep mode. Moreover, as indicated in the empirical studies (e.g., [3]), the energy consumed by network interfaces in the idle state dominates the total energy consumption. These facts lead to several proposals that put the radio into the sleep mode in the lack of communications activities. The objective is to minimize the energy consumption with a minimal impact to the network performance in terms of end-to-end latency, throughput and routing latency. The challenge lies in choosing appropriate times to switch the ratio to the sleep mode and back to the idle mode.

IEEE 802.11 power saving mode (PSM) is a representative of the above power saving mechanisms. All the nodes in the network are time synchronized by a beacon mechanism. They wake up periodically and check whether or not they need to transmit or to receive packets. If not, they go back to sleep and thus conserve energy. As the nodes in the sleep mode can neither transmit nor receive packets, packets that are destined for a sleeping node have to wait for the next beacon interval

(BI) to be transmitted. This is dubbed the *wake-up latency*, a major factor for packet delay. The effect of wake-up latency is especially pronounced when the inter-packet interval is so large that there is no pipeline effect. In this case, a packet can only traverse one hop in a BI, and a router has to wake up twice for one packet – one to receive the packet and the other to forward it.

To deal with this issue in IEEE 802.11 PSM, we propose a novel and complementary mechanism, called *Link-Indexed Statistical traffic Predictor (LISP)*. Essentially LISP employs a simple, light-weight traffic prediction method and enables each node to seek the inherent relation between ATIM_ACKs and incoming traffic. Once such a relation is identified a node en route stays awake in the BI in which a packet is anticipated to arrive, thus bridging a "freeway" for the packet to rapidly traverse the route. In this manner, a packet can travel from the source to the destination within one BI ideally. Meanwhile, the number of duty cycles is minimized and more energy is conserved. As will be elaborated on in Section III, as compared to IEEE 802.11 PSM that consumes energy of intermediate nodes much more than that of the source and destination nodes, LISP also balances energy consumption over the nodes en route. LISP differs from previous work [12], [2], [14] in that we do not trade energy for better end-to-end performance at light to moderate traffic load, i.e., we reduce the end-to-end delay without consuming more power. Instead more energy is saved because of the reduction in duty cycles. We conduct analytical and simulation studies. The impact of various factors, including traffic load, number of hops, ATIM window size and packet sizes are investigated using a tandem topology. In a general scenario simulated, LISP demonstrates the improvement over PSM in the end-to-end delay by 65-75%, and an increasement of the energy efficiency over IEEE 802.11 with and without PSM by 6% and 178%, respectively.

The rest of the paper is organized as follows. In Section II, we provide a detailed summary of existing work. Then we elaborate on LISP in Section III, and conduct the performance analysis in Section IV. Section V discusses some implementation issues. Following that, we present simulation results in Section VI. Finally we conclude the paper in Section VII.

II. Background and Related Work

A. Overview of IEEE 802.11 PSM

IEEE 802.11 provides power management for both PCF and DCF. As we consider in this paper wireless ad hoc networks (that employ DCF), we focus on IEEE 802.11 PSM for DCF unless specified otherwise. In IEEE 802.11 PSM, a node may operate in one of the two modes: Active mode (AM) and Power Save mode (PSM). All the nodes are synchronized in time by a beacon mechanism. The time is divided into beacon intervals (BIs). A BI begins with an ATIM window. For notational convenience, we name the remaining time following the ATIM window in a BI the *DATA window*. All nodes wake up periodically at the beginning of every BI and stay awake through the ATIM window. During that time, if a node has a packet destined for a receiver node, it will send an ATIM packet and the addressed receiver node confirms with an ACK. Such an ATIM-ACK exchange ensures both the sender and the receiver will stay awake in the DATA window in which the packet will be transmitted using the conventional RTS-CTS-DATA-ACK floor acquisition mechanism. On the other hand, if a node neither has a packet to send nor receives an ATIM packet, it goes to sleep at the end of the ATIM window by putting its radio into the sleep mode.

B. Related Work

A number of mechanisms have been proposed to improve the performance of IEEE 802.11 PSM. In this subsection, we summarize existing work and roughly classify them into three categories:

Mechanisms that tune the ATIM window size: Woesner *et al.* [11] investigate, via simulation, the optimal ratio of the ATIM window size to the beacon interval, and suggest the value of approximately 1/4. Jung and Vaidya [4] propose a mechanism of adjusting the ATIM window size dynamically to accommodate varying traffic loads. Tseng*et al.* [10], on the other hand, address the problems caused by in-synchronization and propose asynchronous wake-up. In spite of all the efforts, the problem of prolonged end-to-end delay as a result of PSM remains.

Mechanisms that center on the notion of virtual routing core: Geographic adaptive fidelity (GAF) [12] and Span [2] are two representative mechanisms. GAF partitions the network into geographical rectangular grids. All the nodes in one rectangular grid can communicate directly with nodes in adjacent grids. A leader is elected in each grid to be active, while all the other nodes in the grid can be put into sleep. The leader election scheme in each grid takes into account of battery usage at each node, and a sleeping node wakes up periodically to attempt to elect itself as an active node. To support mobility, in the mobile adaptation version of GAF (GAP-ma), each node estimates the time at which it expects to leave its grid (based on its current speed obtained from GPS and grid size). The simulation study shows that GAF extends the network lifetime by 30-40%.

Span defines a virtual core as a connected dominating set. Nodes make local decisions on whether they should sleep or join a forwarding backbone as a coordinator. The nodes that choose to stay awake and maintain network connectivity/capacity are called coordinators. The rule for electing coordinators is that if two neighbor nodes of a non-coordinator nodes can neither directly communicate with each other nor through one or two coordinators, then this node volunteers to be a coordinator. The information needed for electing oneself as a coordinator is exchanged among neighbors via HELLO messages. The coordinator

announcement is broadcast, based on a delay interval reflecting the "benefit" that each neighbor will perceive and taking into account of the total energy available. The authors then investigated the performance of Span combined with a simple geographic routing protocol and reported power saving by a factor of 3.5 or more.

Mechanisms that are based on the notion of wake-up-on-demand: On-demand power management (on-demand) [14], STEM [8] and S-MAC [13]) are representatives in this category. The work that comes closest to ours is on-demand. In on-demand every node transits to AM when it receives/forwards routing-related packets (e.g., route discovery/reply packets in DSR or AODV) and/or data packets, and remains awake for an extended period assuming future communication activities. A soft-state timer is maintained at each node, and is set/reset upon arrival of routing/data packets. Upon timeout, the node then switches back to the power saving mode. The time-out values used are, however, pre-specified parameters, typically in the range of 2-5 seconds. As a result, for a connection with a continuous packet stream, all nodes en route are likely to be in the active mode, rending an end-to-end delay that is similar to the case without PSM.

STEM and S-MAC target for sensor networks, and share the similarity of periodic sleep and wake-up with IEEE 802.11 PSM. They embrace the wake-up-on-demand idea in their respective ways. STEM uses a second control channel to monitor the network status. S-MAC introduces a message-passing mechanism to allow packets to be sent in a burst.

Both virtual routing core-based mechanisms and on-demand mechanisms share the similarity that both attempt to keep a subset of nodes awake in the course of communication activities. The major difference, however, lies in that the former maintains an always active routing backbone and performs well when there exists a substantial amount of traffic to amortize the overhead of building/maintaining the backbone. The on-demand mechanisms, on the other hand, react in the presence of communication activities. Its performance relies heavily on the appropriate setting of timer values.

In contrast to the above two categories of mechanisms, LISP enables nodes en route to enter the active mode *only* in beacon intervals in which packets are *anticipated* to arrive. That is, nodes respond to communication activities at the packet level, rather than at the connection level (as was done in on-demand). By employing a simple, light-weight traffic prediction method and performing power management at a finer time granularity, LISP further reduces the energy consumption incurred in order to improve the end-to-end packet delay.

III. Link Indexed Statistical Traffic Predictor

A. Overview of the Basic Mechanism

As discussed in Section I, the major factor that leads to the increase in the end-to-end delay in ad hoc networks with PSM is the wake-up latency. Ideally, a node should stay awake and forward packets in the same BI in which a packet will arrive. However, this may not be possible without the detailed information provided by the upper layers. On-demand, for example, simply enables all nodes en route to stay awake during the duration of a connection. In this section, we discuss how we perform power management at a finer time granularity by using a light-weight prediction mechanism.

The key idea in the proposed traffic prediction mechanism is to take advantage of the correlation between the ATIM-ACK exchange of upstream nodes and the incoming traffic. We use the following example to illustrate the idea.

Example 1: Consider a single connection that traverse the route: node 1 $\rightarrow$ node 2 $\rightarrow$ node 3. Suppose a packet is generated at node 1 at time t_0. As shown in Fig. 1, node 1 has to wait for the next BI, BI_1, to initiate an ATIM-ACK exchange with node 2 and then to forward the packet in the DATA window. Similarly, node 2 has to wait for the next BI, BI_2, to forward the packet to node 3. Note that the event that a data packet will arrive at node 3 (indicated by an ATIM packet in BI_2) is preceded by the event that an ATIM_ACK packet was sent by node 2 in BI_1. Note also that the latter event can be naturally overheard by node 3, as node 3 is the next hop en route. Therefore, if node 3 can judiciously interpret the ATIM_ACK packet as an indicator of incoming traffic, it can stay awake through BI_1. In this manner, the packet can be transmitted to node 3 in BI_1, and the wake-up latency can be greatly reduced.

To realize the above idea, we have to consider three issues. The first issue is how node 3 learns an overheard ATIM_ACK is indeed the preamble of future packet arrival. Each node snoops the header of overheard packets. If it overhears an ATIM_ACK packet addressed for some other node in a BI, and receives an ATIM in the subsequent BI, it recognizes the correlation. Next time when it overhears an ATIM_ACK packet again, it takes that as an indication of incoming traffic and herein stays awake through the BI. On the other hand, if such a conjecture is incorrect and the node receives no packets in the BI, then it "erases" the recent history and learns from scratch.

The second issue is concerned with how an upstream node (e.g., node 2 in Example 1) knows its downstream node (e.g., node 3 in Example 1) has detected the correlation and is awake in the current BI, so that it can forward the packet to the downstream node. The third issue is that in the case of more than 2 hops, how downstream nodes perform prediction in the lack of ATIM-ACK exchange. These two issues are resolved by having a node i transmit a pseudo-ACK packet when it overhears the ATIM_ACK packet. This pseudo-ACK packet notifies node i's upstream node of its willingness to stay awake in the current BI. Moreover, this pseudo-ACK packet is taken by node i's downstream node as if it were an ATIM_ACK packet (in the sense that it indicates packet arrival in the immediate future). In this manner, a "chain" effect can be triggered. For notational convenience,

we will henceforth term both ATIM_ACK packets and pseudo-ACK packets as *traffic indicators* as both of them serve the purpose of indicating future traffic.

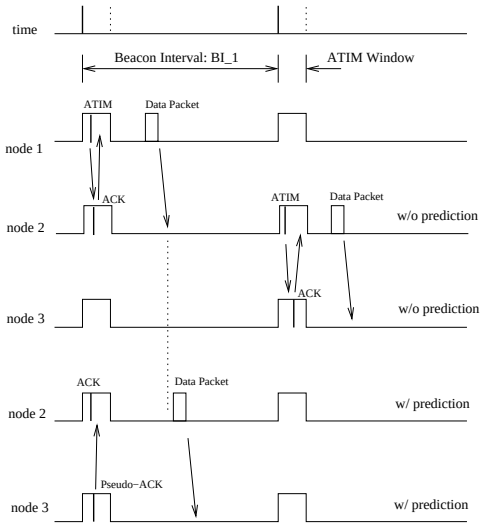


Fig. 1. An example that demonstrates the basic idea of LISP. A single connection traverses the route: node 1→ node 2 → node 3.

B. Detailed Description of the Basic Mechanism

In this subsection we give a formal description of how LISP operates. There are two phases in the traffic prediction process: learning and prediction. In the learning phase, a node makes a conjecture and attempts to confirm it. Specifically, if in the ATIM window of a BI, a node A overhears from its neighbor node B an ATIM_ACK packet addressed for some other node, node A conjectures itself to be the next hop node. If in the next BI, node A indeed receives an ATIM packet from node B, it confirms the conjecture and will enter the prediction phase. Otherwise, node A resets its state and starts all over again. In the prediction phase, node A predicts that node B will forward one or more packets to itself and will stay awake through the current BI. Meanwhile, node A transmits a pseudo-ACK packet announcing this fact. All of node A's neighbors that receive this pseudo-ACK packet will take it as if it were a ATIM_ACK packet and run the traffic prediction process accordingly. Node A is said to be in the *Temporary Active Mode*, or *TAM*, when it stays active in the current BI and its neighbors can forward packets to it even if they have not performed an ATIM-ACK exchange in the ATIM window of the current BI. The TAM expires automatically when the next BI commences and the node switches back to its previous mode.

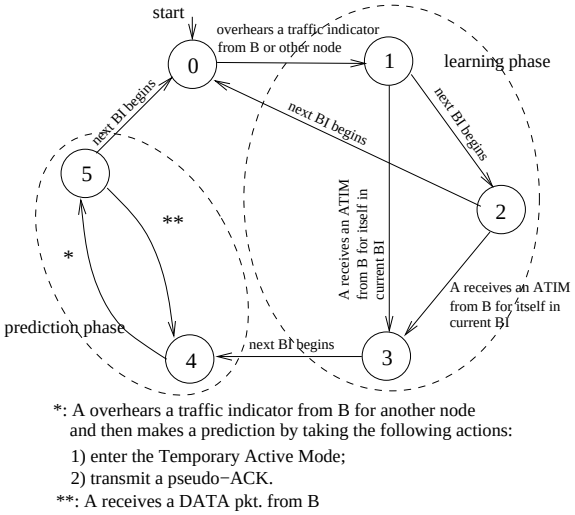


Fig. 2. Finite state machine that describes the operations of LISP.

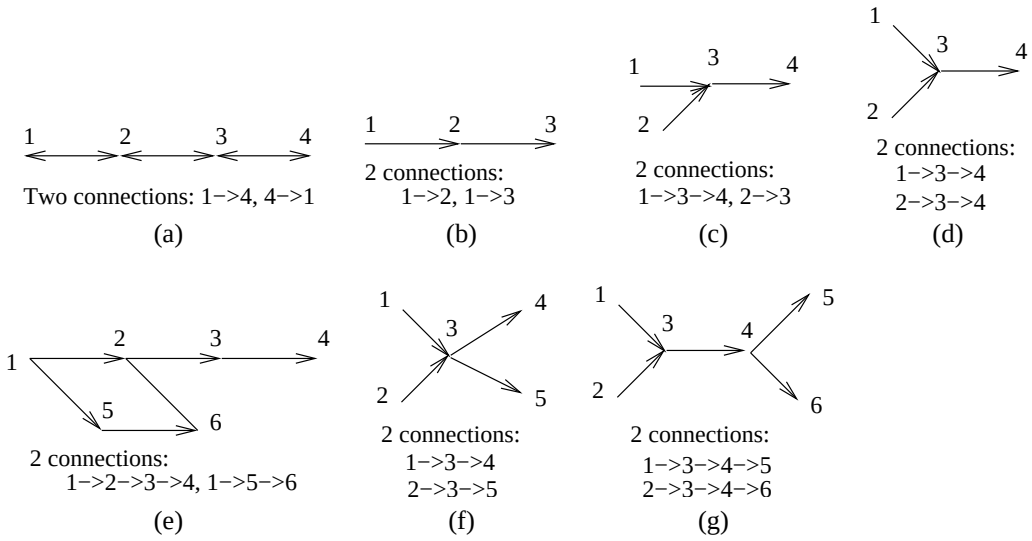


Fig. 3. Possible traffic patterns in the presence of multiple connections.

The above operations can be described by a finite state machine (Fig. 2). Every node runs a finite state machine for each overheard neighbor. We define such a state as the *node-indexed state* (*NI-state*). The state starts at 0. Node A snoops every packets it hears. If in the current BI it overhears from a neighbor node B a traffic indicator addressed for another node, it enters state 1. When the next BI commences, node A transits to state 2, waiting for an ATIM packet from node B to confirm the conjecture. If this is indeed the case, node A enters state 3; otherwise, when the next BI commences, the state is reset. In the former case, the prediction phase begins in the next BI since node A will stay awake in the current BI anyway. Node A transits from state 4 to state 5 when it overhears a traffic indicator from node B and transits from state 5 back to state 4 when a data packet arrives in the same BI. The prediction phase ends in state 5, if node A does not receive any data packets from node B. Node A returns to state 0 and the traffic prediction process repeats.

Note that there is a transition from state 1 to state 3. This occurs when node A overhears a traffic indicator and then receives an ATIM or data packet in the same BI. This can take place when the traffic load is no less than one packet per BI.

C. Extension to Multiple Connections – Node-Indexed, Link-Indexed or Connection-Indexed?

The basic mechanism may not operate correctly in the presence of multiple connections, as connections may share nodes or links on their routes, and a node may receive/forward packets from different connections in an interleaving fashion. As a result, the correlation between traffic indicators and arrivals of packets for one connection has to be differentiated from those for other connections. For example, consider the scenario in Figure 3 (b). If node 3 does not differentiate the states kept for connections $1 \rightarrow 2$ and $1 \rightarrow 2 \rightarrow 3$ and if the packets of the two connections arrive at node 2 in an interleaving fashion, node 3 may not be able to make an accurate prediction. (Similar conclusions can be reached for other scenarios except (a) in Fig. 3.) This implies indexing state by node ID is not sufficient.

To resolve the above issue, we consider two alternatives: (i) *Link-Indexed state* (*LI-state*): the state indexed by a link identified by its two end points: $\langle$node 1, node 2$\rangle$; and (ii) *Connection-Indexed state* (*CI-state*): the state indexed by the source and destination of a connection: $\langle$src, dst$\rangle$. To evaluate the effectiveness of the three state representations, we enumerate a number of representative traffic patterns in Fig. 3 and give in Table I whether NI-states, LI-states and CI-states suffice under these cases. As shown in Table I, NI-states do not operate well in the multiple connection case. The LI-states fail at nodes at which connections diverge. For example, in Fig. 3 (g), the two connections share a common link $3 \rightarrow 4$. Node 5 cannot tell from the ATIM_ACK packet sent by node 4 to node 3 which node the incoming packet will be destined for (nodes 5 or 6). Consequently, node 5 may make inaccurate prediction and determine to reset the state, resulting in a large end-to-end delay for the next packet.

The CI-states perform best. As long as a node receives a traffic indicator with the connection information enclosed, it can tell whether it should keep awake. However, with a closer investigation, we find that indexing states with connections may not always be feasible. First, the ATIM_ACK operation is basically link-oriented. As IEEE 802.11 PSM states, a node only needs to send one ATIM packet to the intended receiver in a BI even if it has multiple data packets from different connections. Indeed, multiple ATIM requests would consume bandwidth and cause contention unnecessarily. Second, the connection identification information is only available at the network and higher layers. To obtain the source and the destination information, the link level mechanism has to look into the network layer header of a packet, thus introducing cross-layer overheads. Third, to include

TABLE I

PERFORMANCE OF THE THREE STATE REPRESENTATIONS UNDER TRAFFIC PATTERNS GIVEN IN FIG. 3.

Effectiveness	a	b	c	d	e	f	g
NI-states	OK	F	F	F	F	F	F
LI-states	OK	F	OK	OK	OK	OK	F
CI-states	OK	OK	OK	OK	OK	OK	OK

the connection information in each ATIM and ATIM_ACK packets one has to increase the packet size. In contrast, one needs only to attach the receiver's address in the ATIM_ACK packet in the case of the LI-states.

With both the pros and cons taken into consideration, the LI-states are perhaps the best candidate in terms of effectiveness and feasibility. To make up for its deficiency, we introduce in Section III-D a statistical technique attempting to profile the traffic with the maximum likelihood. To reflect the fact that the statistical technique is used, we name the enhanced mechanism the *Link-Indexed Statistical traffic Predictor* (*LISP*).

D. Full Fledged Mechanism – LISP

We enhance the basic mechanism in two aspects. First, the state is link-indexed. A node maintains a state for each overheard link. Second, the prediction is not made solely on the traffic indicator, but also on the statistical result computed from the past traffic history. To this end, after a node enters the prediction phase, it records the correlation between a traffic indicator and the packet arrival. If a packet does arrive in the immediate near future, the node records a '1' for the link; otherwise, a '0.' The number of records kept for each state is at most K, where K is a pre-determined constant ($K = 8$ for example.) At the time of making prediction, the node calculates the percentage of positive correlations in the past, i.e., $p = \frac{\text{number of '1' in } M \text{ records}}{M}$, where M is the number of records kept so far. In some sense *p* measures the credibility of traffic indicators. A random number r that is uniformly distributed in [0, 1] is generated and compared against p. If $r \leq p$, the node predicts that a packet will arrive in the current BI and enters the TAM if it is not in yet; otherwise, it predicts the opposite and does nothing.

The revised finite state machine that describes the full fledged mechanism is given in Fig. 4. In particular, Fig. 4 (a) gives the finite state machine to be run for a LI-state $\langle$B, C$\rangle$ at node A. The state transitions are similar to those in Fig. 2. The transition from state 4 to state 5 takes place only when a positive prediction is made. Moreover, in the prediction phase the state will not be reset unless all the records are zeros. This is indicated by an arrow from state 4 pointing to state 0 and is triggered by the event of all records becoming zeros.

The other finite state machine given in Fig. 4 (b) is run for the associated *record* state, which is used to generate a current record. There are three typical transition paths: 0→1→0, 0→1→2→0 and 0→1→2→3→0, recording the correlations in the three cases illustrated, respectively, in Fig. 5. More complicated transition paths exist, e.g., 0→1→2(→3→2)→0 and 0→1→2(→3→2)→3→0, where ($\rightarrow i \rightarrow j$) denotes the transition can repeat one or multiple times. They are simply combinations of the three cases.

E. Fault Tolerance

In addition to the case of presence of multiple connections, traffic prediction may not be accurately made in the case of collisions. A node may miss its chance of overhearing a traffic indicator when there are collisions. In general, ATIM_ACK packets are more immune to collisions as the nodes hearing the CTS (in the RTS/CTS floor acquisition mechanism) have performed necessary backoffs. However, it is more likely that pseudo-ACK packets incur collision with other on-going transmissions (e.g. beacon messages). The aftermaths are twofold: first, from the viewpoint of a downstream node, if it misses a traffic indicator in the learning phase, it will wait longer to enter the prediction phase and the first few packets incur larger end-to-end delays. If the node misses a traffic indicator in the prediction phase, it fails to perform the prediction and the packet sent in the current BI incurs a larger end-to-end delay. The penalty in the second case is more severe. Suppose the downstream node enters the prediction phase and transmits a pseudo-ACK packet, but the upstream node misses it. Then in the following DATA window, even though the downstream node is actually awake, the upstream node will not forward packet(s) to it. This not only delays the delivery of the packet but also lures the downstream node to reset its state because it keeps awake but receives nothing.

Fortunately the statistical technique can also be used to mitigate the negative effect of inaccurate prediction caused by packet collision.

IV. PERFORMANCE ANALYSIS

A. Effect of Wake-up Latency

As pointed out in Section III, the wake-up latency is the most important factor that degrades the end-to-end performance of PSM. It increases the end-to-end delay in the scale of BIs even under the light traffic case. In the case of heavy traffic, the

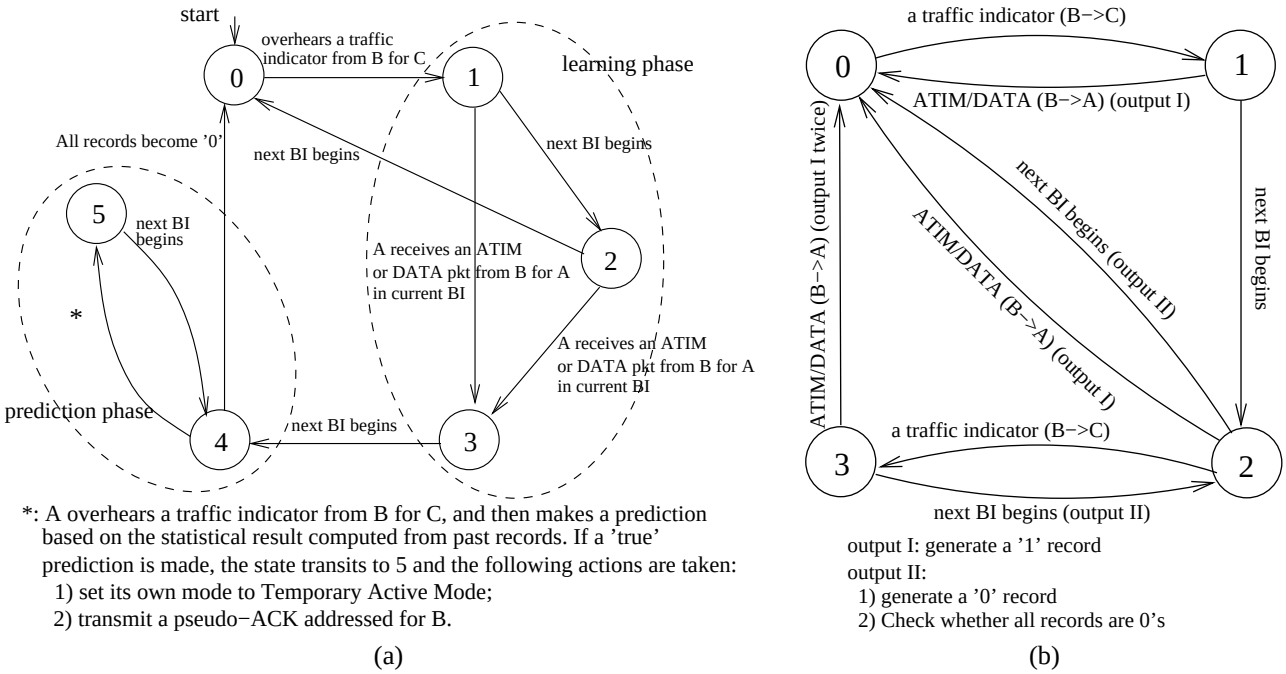


Fig. 4. Finite state machines running at node A for link ⟨B, C⟩

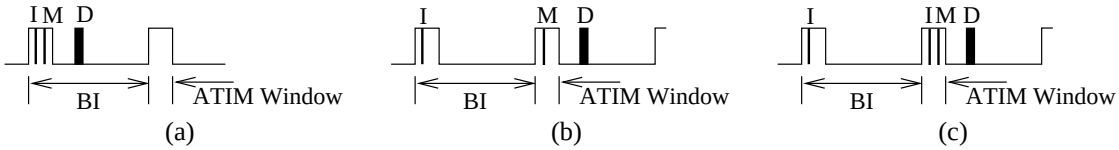


Fig. 5. Three cases to be considered in the finite state machine given in Fig. 4 (b) for the associated *record* state. I – a traffic indicator, M – an ATIM packet, D – a data packet.

effect of the wake-up latency couples with the contention and queuing delays, and may significantly degrade the performance. To analyze the impact of the wake-up latency in isolation of other factors, we study in this section the performance of IEEE 802.11 with (termed as PSM) and without PSM (termed as nonPSM) and LISP under a simple yet representative scenario. For notational convenience, we define the notations to be used throughout the analysis in Table II.

1) Network Model and Performance Metrics: We consider a wireless network with a single connection that traverses H hops: $n_0 \rightarrow n_1 \rightarrow ... \rightarrow n_H$. Wireless nodes are assumed to be time synchronized. The per hop MAC delay for packets of the same size is assumed to be the same and denoted by ΔP. Note that ΔP includes the transmission time of RTS, CTS, DATA and ACK packets, the inter-frame time and the propagation time. In general, *BI* is much larger than ΔP. We also assume the length of an ATIM window, T_{ATIM} is large enough so that all the nodes en route can make prediction within a time interval of T_{ATIM}.

We consider the case in which inter-packet intervals are large enough to avoid the pipeline effect, i.e., every node receives or forwards at most one packet in a BI. This is, in general, true when the inter-packet interval is larger than one BI. Hence we assume the packet generation rate of the connection is $\lambda \leq 0.5BI^{-1}$. Under this assumption, the effect of contention is negligible and the wake-up latency dominates the end-to-end delay.

We will derive the end-to-end delay, D, and the total power consumed under the scenario described above. To quantify the total power consumed, we define a new metrics, *duty cycle ratio* R, as the number of duty cycles over the total number of BIs throughout the duration of a connection, where a duty cycle is a BI in which the node stays awake. Note that R is a good metric of power consumption for any energy model in which the power consumed in the idle state is comparable to that in transmitting and receiving. It can measure the percentage of power consumed with and without power management.

BI	Length of a BI.
T_{ATIM}	Length of an ATIM window.
ΔP	Per hop MAC layer delay without PSM.
λ	Packets generation rate of a connection (in terms of number of packets per BI).
t_i	One hop delay of a packet from node n_{i-1} to node n_i.
t_g	Given that a packet is generated at the source node at time t, t_g is the relative offset to the beginning of the current BI. t_g is assumed to be uniformly distributed in $[0, BI]$.
D	End-to-end delay.
N	Total number of BIs throughout the duration of a connection.
N'	Number of duty cycles of a node throughout the duration of a connection.
R	Duty cycle ratio, $= N'/N$.

TABLE II
NOTATIONS USED THROUGHOUT THE ANALYSIS.

2) IEEE 802.11 without PSM (nonPSM): Under the above assumption, the average duty cycle ratio and the end-to-end delay of nonPSM can be expressed, respectively, as

$$\overline{D}_{nonPSM} = H\Delta P, \tag{1}$$

$$\overline{R}_{nonPSM} = 1, \tag{2}$$

where H is the number of hops of the route.

3) IEEE 802.11 PSM: Under IEEE 802.11 PSM, it takes one BI for a node to receive or to forward a packet. Therefore, it takes one BI for the source (the destination) to send (receive) a packet, and two BIs for intermediate nodes to receive a packet and to forward it. For a connection that lasts for N time units (note that the time is normalized with respect to BIs), a total of λN packets are generated, and

$$N'_{src} = N'_{dst} = \lambda N, \tag{3}$$

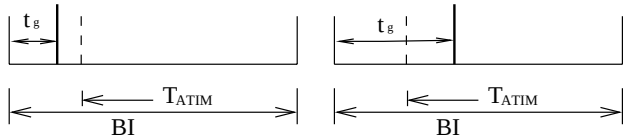
$$N'_{\textit{intermediate nodes}} = 2\lambda N. \tag{4}$$

Let the average number of duty cycles over all nodes en route be denoted as $\overline{N'}$. Then we have

$$\begin{aligned} \overline{N'}_{PSM} &= \frac{1}{H+1}(2\lambda N + (H-1)\times 2\lambda N) \\ &= \frac{H}{H+1}\times 2\lambda N \end{aligned} \tag{5}$$

The average duty cycle ratio is thus

$$\overline{R}_{PSM} = 2\lambda\frac{H}{H+1}. \tag{6}$$



(a) $t_g < T_{ATIM}$ (b) $t_g \geq T_{ATIM}$

Fig. 6. The time at which a packet is generated in a BI.

Next we derive the amount of time it takes for a packet to traverse one hop. Given the definition of t_g and t_i in Table II, we consider two cases where t_g may fall in a BI (Fig. 6): If $t_g < T_{ATIM}$, the packet may be transmitted in the current BI; otherwise, it has to wait for the next BI to be transmitted. In the former case, $t_1 = T_{ATIM} - t_g + \Delta P$, while in the latter case $t_1 = BI - t_g + T_{ATIM} + \Delta P$. After the first hop, when the packet arrives at node n_{i-1} ($i > 1$), it has to wait for next BI to be forwarded since n_i is asleep. Hence $t_i = BI - (T_{ATIM} + \Delta P) + (T_{ATIM} + \Delta P) = BI$ for $i > 1$. The end-to-end delay

of a packet is therefore

$$D = \sum_{i=1}^{H} t_i = t_1 + (H-1)BI, \quad \text{and} \tag{7}$$

$$\overline{D} = E[D] = E[t_1] + (H-1)BI. \tag{8}$$

Under the assumption that t_g is uniformly distributed in $[0, BI]$, the expectation of t_1 can be expressed as

$$\begin{aligned} E[t_1] = {} & P(t_g < T_{ATIM})E[t_1|t_g < T_{ATIM}] \\ & + P(t_g \geq T_{ATIM})E[t_1|t_g \geq T_{ATIM}], \end{aligned} \tag{9}$$

where

$$P(t_g < T_{ATIM}) = \frac{T_{ATIM}}{BI}, \tag{10}$$

$$\begin{aligned} E[t_1|t_g < T_{ATIM}] &= E[T_{ATIM} - t_g + \Delta P|t_g < T_{ATIM}] \\ &= \frac{T_{ATIM}}{2} + \Delta P, \end{aligned} \tag{11}$$

and

$$\begin{aligned} E[t_1|t_g \geq T_{ATIM}] &= E[BI - t_g + T_{ATIM} + \Delta P|t_g \geq T_{ATIM}] \\ &= \frac{BI}{2} + \frac{T_{ATIM}}{2} + \Delta P. \end{aligned} \tag{12}$$

Combining Eqs. (10)-(12) with Eq. (9), we have

$$E[t_1] = \frac{1}{2}BI + \Delta P. \tag{13}$$

Finally, we have

$$\overline{D}_{PSM} = (H - \frac{1}{2})BI + \Delta P. \tag{14}$$

4) LISP: The ATIM_ACK from the source triggers the traffic prediction operation at the next hop, and pseudo-ACKs are transmitted sequentially one by one till the destination. Hence when the packet arrives at the next hop of the source node, all the downstream nodes keep awake, and their respective upstream nodes are notified of this fact. As a result, the packet reaches the destination node in one BI. Also, the number of duty cycles of any node en route is

$$\overline{N'} = N' = \lambda N, \tag{15}$$

and

$$\overline{R}_{LISP} = \lambda. \tag{16}$$

The end-to-end packet delay under LISP is

$$D = t_1 + (H-1)\Delta P \tag{17}$$

By Eq. (13), we have

$$\overline{D}_{LISP} = \frac{1}{2}BI + H\Delta P \tag{18}$$

5) Comparisons among IEEE 802.11 with and without PSM and LISP: Comparing Eqs. (2), (6) and (16), we have $\overline{R}_{nonPSM} > \overline{R}_{PSM} > \overline{R}_{LISP}$. When the traffic rate λ is smaller than one packet every other BI, the average active time of a node under PSM is approximately $2\lambda\frac{H}{H+1}$ of that without PSM. This explains why PSM can save power. Note, however, that the amount of saving decreases when the number of hops increases. LISP further reduces the active time to approximately λN regardless of the number of hops.

PSM conserves power consumption at the expense of larger end-to-end delay. By Eqs. (1), (14) and (18), we have $\overline{D}_{nonPSM} < \overline{D}_{LISP} < \overline{D}_{PSM}$. The dominant component of $\overline{D}_{PSM}$ (note that typically $BI >> \Delta P$) is the product of BI and the number of hops, while the dominant component of $\overline{D}_{LISP}$ is $BI/2$, independent of the number of hops (ideally) and is comparable to one BI. This is a significant improvement, especially considering the fact that LISP also conserves more energy at the same time.

We also note that LISP makes all the performance improvement without much overhead. The computation overhead incurred at a node is that of running two finite state machines for each overheard link. A few bytes are needed for each node to maintain the states. The communication overhead incurred is to transmit a pseudo-ACK in the case of making a positive prediction. The

impact of transmitting pseudo-ACK packets in an ATIM window on other on-going traffic is not significant, as pseudo-ACK packets are small. Moreover, these pseudo-ACK packets actually play the role of receiver-polling and hence the upstream nodes do not have to initiate ATIM-ACK exchanges.

B. Effect of Other Factors

In more general scenarios where the traffic load is heavy and multiple connections exist, the effect of queuing delay, medium access contention may couple with the wake-up latency and degrade the performance of both PSM and LISP. We will investigate their impact by our simulation study under general scenarios (Section VI).

In particular, LISP has its own limitations. First, as discussed in Section III-E ATIM_ACK and pseudo-ACK packets are prone to collisions. Second, as discussed in Section III-C, prediction on the incoming traffic may not be accurately made in the case that connections share common links and then diverge. Although the effect of inaccurate prediction is mitigated by the statistical approach proposed in Section III-D, it still degrades the performance to some extent in the presence of multiple connections.

V. Implementation

We implement LISP using *ns2*. The original code for IEEE 802.11 PSM we used was developped by Span (add ref. link). During the development of LISP, we note several possible variations of the implementation of PSM and their impacts to the performance. As for LISP is directly based on PSM, those impacts can also be observed in LISP. Therefore we make a brief discussion here.

- Passive PSM. At the beginning of every BI, a node will examine whether it has buffered packets and sends ATIM packets for them. Then it will send buffered packets when the current ATIM window ends. After this, it will stay awake through the rest time of the BI, even if there isn't any new arriving packets. Alternatively, the node can choose to goto sleep to save power after transmitting all buffered packets. In this way, the node is passive in participating network activities and therefore we name such a variation *passive PSM*. This passive idea is seen in [15] and is implemented in a wider sense. For example, [15] sends ATIM packets for buffered packets but doesn't track to which nodes ATIM packets have been sent. Packets arrived in the same BI after all the buffered packets destined for the same node(s) are transmitted thus lose the chance to be transmitted.
  Under low traffic load, passive PSM can save more power. However, It results in larger delay under bursty traffic with inter-arrival time smaller than one BI. Moreover, the network capacity is reduced. The network begins to drop packets earlier than PSM. These are observed in our simulation study. Typically, the bandwidth is limited to 1-2 packets per BI regardless of the packet size (we simulate with the packet sizes varying from 100 to 1K bytes).
- Handling a packet cross the boundary. With the introduction of periodic wake-up, the transmission of a packet (RTS-CTS-DATA-ACK) may cross the beginning of a BI. From the viewpoint of the sender node, the corss point may occurs 1)after transmitting RTS but before transmitting the DATA packet; or 2)after transmitting the DATA packet but having not received the ACK packet yet. (Note that failure of transmitting the RTS or DATA packet will result in a time-out at the sender node. The node will examine whether it is in the ATIM window and re-enqueue the packet in the buffer if yes.) Since the CTS and ACK packets are allowed in the ATIM windows, in case 2 the transmission of the packet can still succeed. In case 1, the sender node may still receive the CTS packet if the RTS packet is successfully received by the addressed node. However, the sender node will suspend transmitting the DATA packet for the moment and turn to perform the actitivities defined in the ATIM window. Since this packet has been dequeued from the MAC layer buffer, it becomes a pending packet, which has to be carefully handled. If the node simply puts off the transmission till the current ATIM window ends and then resumes by restarting the RTS-CTS-DATA-ACK process, the receiver node may have gone into sleep. This happens when there are no other packets buffered for the same destination node and hence the sender node will not send an ATIM packet for the pending packet. In this case, the sender node will finally fail in receiving the CTS packet and dropps the DATA packet. Meanwhile, an event of link failure will be falsely triggerred at the upper layer routing protocol and incurs unnecessary communication overhead and large delay. This is observed in the implementation of [15]. The simulation study shows that mis-handling of the boundary problem impairs the performance to a large degree once the burstiness of the traffic is close to or exceed one packet per BI.
  There are several ways to address this issue. First, the sender node does not start any new transmission process if the time to the beginning of next BI is not enough to finish the process. Thus there will not be any pending packet whose transmission process will cross the boundary of a BI. When the next BI begins, the sender node will examine the buffer and file an ATIM-ACK exchange for any buffered packets as usual. The second solution is that the receiver node keeps awake in the next BI automatically even if not receiving an ATIM packet from the sender node. This solution is survivable because both the sender and the receiver nodes can get informed from the successful RTS-CTS exchange. Doing so can

save an ATIM-ACK exchange if no other buffered packets are destined to the same node (The avoidance of an ATIM-ACK exchange is still possible even if the case is true. To do this, the node has to add a state variable to record the suspending status. A little more complexity will be involved, however.). The third solution is that in the ATIM window a node not only scans buffered packets but also checks the pending packet to file an ATIM packet(s). The first and third solution shares some similarity in that an ATIM packet will be issued for the packet anyway. The first one might save some power from transmitting the RTS packet. However, it requires extra operations including re-enqueing the pending packet and resetting related state variables. Moreover, if by chance due to imperfect synchronization, the start of a BI in the sender node is earlier than that of the receiver node, the transmission of packet could have been finished successfully but is avoided of the chance in the first solution. Actually, when there are coexistence of PSM-nodes and non-PSM-nodes, this disadvantage is more obvious. Therefore, for favor of less complexity and more chances of transmission, we prefer the third solution to the first one. The second solution also involves of more complexity. Besides, the prediction mechanism that LISP uses depends on an overheard ATIM_ACK exchange. Thus we assumes the third solution in our implementation.

In case of imperfect synchronization, the aforementioned boundary problem can affect LISP in another way. In PSM, one bit in the MAC layer header of every packet is used to indicate the current power management mode: *active* or *PSM*. LISP introduces one more mode as specified in Section III: *TAM*, or Temporary Active Mode. Thus one more bit is needed. Once a node makes a true prediction, it will enter the TAM mode and this is reflected in every packet it transmits in the current BI. Nodes overhearing a packet will update their local records of the sender node's power management status. If the mode is TAM, the record will be reset to PSM automatically when the next BI begins. However, due to imperfect synchronization, nodes may start a BI at different time. The TAM mode, which was one node's power management status in the last BI, may be misinterpreted by other nodes as the node's status in the current BI. We illustrate such a case with an example as shown in Fig. 7. Node A is the upstream hop of node B. Node B is in the TAM mode in BI0. Near the end of BI0, node B receives a DATA packet from node A and accordingly replies with an ACK. At this time, node B hasn't begun the next BI, BI1, so the ACK packet still indicates of the TAM mode. Whilst node A has entered BI1 thus interprets node B will still be in the TAM mode in BI1, which however by chance is not true. Thus even if node A has any packet buffered for node B, it will not file an ATIM packet. The buffered packet thus will be dropped after retrying transmitting RTS to the maximum limit and then an event of link breakage will be triggered. Similar cases can happen if the transmission of a packet is interrupted after the sender node receiving a CTS from the receiver node.

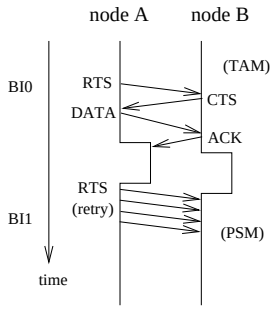


Fig. 7. An example of the boundary problem caused by inaccurate synchronization

The key of the solution to this problem is to let the node be able to identify whether an ACK or CTS packet is sent during the sender node's ATIM window. We handle this problem in the following way. First, we use a bit in the MAC layer header of the ACK packet, which was used to indiciate <more_data>. The bit is set if the ACK packet is sent in the ATIM window, thus avoiding the ambiguity. Besides, during the ATIM window, a node only spys BEACON/ATIM/ACK packets to learn neighbors' power management status.

The experience we have earned from developing PSM and LISP shows that careful implementation is critical to achieve expected network performance and to avoid some traps.

VI. Simulation Study

Comprehensive simulations are conducted to study the performance of LISP, which is compared against PSM, IEEE 802.11 without PSM (nonPSM) and on-demand [14]. Six metrics are considered: *end-to-end delay*, *energy efficiency*, *packet delivery ratio (or normalized throughput)*, *energy efficiency throughput product*, *energy cost per bit delay product* and *duty cycle ratio*, denoted by D, EE, η, $EETP$, $EPBD$ and R, respectively. *Energy efficiency* (bits/J) is the end-to-end throughput (bits) over energy (Joule) consumed in the duration of a simulation run. *Energy efficiency throughput product* is the product of *energy efficiency* and *normalized throughput*. *Energy cost per bit delay product* is the product of the amount of energy used to delivery a

bit and the end-to-end delay, or D/EE. The first three metrics evaluate the network performance (D and η) and the effectiveness of power saving (EE) indivudually. While the fourth and the fifth metrics take both into consideration, trying to capture a curve of tradeoff between the two from two different aspects. $EETP$ not only indicates the energy efficiency for the amount of traffic delivered but also exhibits the capability of delivering packets. $EPBD$, proposed by R.Zheng, et.al. to study the tradeoff between energy saving and delay. The last metrics, R, measures how much time an approach put nodes into sleep. There may be redundance in these metrics, through simulation we will try to locate the most representative ones.

Various factors are studied: traffic load, number of hops, the ATIM window size, and packet size (synchronization, maybe later). To signify the impact of each factor has on the performance, we first study a tandem topology of $H + 1$ nodes (that line up at an equal distance of 200m) with a CBR connection from the first node to the end node. Only neighboring nodes can communicate with each other. Then we consider a general scenario with nodes randomly distributed in the field.

The common setup of all simulations are as follows. All the nodes communicate with each other with half-duplex radio, and have a uniform communication range of 250m. The energy model assumes the one given in in [3], i.e., it takes 1400mW, 1000mW, 830mW and 130mW for a node to transmit, receive, stay idle and sleep, respectively. The energy consumed for switching between the idle and sleep modes is assumed to be negligible and not counted in the simulations. The dynamic source routing protocol (DSR) is used. Long live CBR traffic is carried from the source to the destination nodes. All packets have the uniform size of 1K bytes. Each simulation run lasts for 500 seconds, and each data point is an average of 10 simulation runs.

A. Performance in a Tandem Wireless Network

1) Effect of Traffic Load: We first investigate the effect of traffic load, one of the most important and concerned factors, on the performance. This is studied in a tandem wireless network as described above. A number of scenarios are simulated with a large range of traffic load and different number of hops H. Fig. 8 presents a set of results when $H = 4$. The packet generation rate λ is ranged from 0.1 to 8 packets/BI. Comparing the performance of the four mechanisms in regard to each metrics, we have the following observations.

- LISP makes a great improvement over PSM by suppressing the end-to-end delay to around one BI. As presented in Fig. 8 (a), the end-to-end delay under LISP is around 0.24-0.29 of that of PSM before the network capacity is reached.
- LISP is the most energy efficient one of all the compared mechanisms under small and moderate traffic load as shown in Fig. 8 (b). Only after the traffic load becomes heavy, $\lambda > 2$ packets/BI in this case, PSM achieves the highest energy efficiency. LISP still outperforms than nonPSM and on-demand before the network capacity is reached though. This change of trend is reflected in Fig. 8 (f) as well. When $\lambda < 2$ packets/BI, LISP has the lowest duty cycle ratio. As λ increases, the pipeline effect becomes more obvious, which benefits both PSM and LISP. However, the mechanism of LISP tries to keep every node on the route awake in a BI once there are traffic; while PSM tends to pile up packets at a node and to forward them in the succeeding BI. Thus LISP carries the traffic in a more smooth way, but incurs more duty cycles in this case. When λ exceeds the network capacity, 3 packets/BI in this case, all the PSM-based mechanisms consumes more energy than nonPSM. This is because activities in the ATIM windows are avoided in nonPSM and that the reduced capacity in PSM-based mechanisms may result in more retransmissions in the congestion state.
- LISP has no negative impact of the throughput before the network capacity is reached. Actually all the three PSM-based mechanisms maintain 100% packet delivery ratio as long as the network is not overloaded as shown in Fig. 8 (c). However, when the network is overloaded, all the three PSM-based mechanisms drops more packets than nonPSM. Similar trend is observed in other scenarios with different number of hops and different size of the ATIM window. In the presented case with H$H = 4$, LISP drops more packets than the other two PSM-based mechanisms. In simulations with fewer number of hops, LISP drops comparable or less packets. It indicates that the prediction chain is prone to longer route and more pseudo-ACKs needed for longer route may result in the ATIM window congested more serverly under heavy traffic load.
- Fig. 8 (d) plots the curves of the energy efficiency throughput product as λ increases. These curves reflect the overal achievement with both the energy efficiency and the throughput considered. As for the curves of normalized throughput (Fig. 8 (c)) are flat before $\lambda = 3$packets/BI and the curves of energy efficiency (Fig. 8(d)) begin to become flat after $\lambda = 3$ packets/BI, the product of the two actually plays approximately as a substitute of the two. This suggests that in PSM-based mechanisms there is no tradeoff between energy efficiency and throughput – when PSM has advantage of energy efficiency, it doesn't lose throughput; when PSM loses that advantage, the throughput gets worse as well. This claim may not apply widely. For example, the buffer at a node has to be large enough to hold packets during the sleep mode.
- Fig. 8 (e) is a combination of Fig. 8 (a) and (b). The figure shows that before the network is overloaded nonPSM has the lowest EBDP, on-demand and LISP come the next in order, and PSM has the highest EBDP. It demonstrades that the benefit of energy saving cannot reverse the loss in end-to-end delay for PSM-based mechanisms. This is because the periodic wake-up increases end-to-end delay in scale of BI, which is tens of that of nonPSM. While the energy saving can be only at most several tens of percentage in the simulated range. This metrics doesn't provide much extra instructive information and cannot replace (a) and (b), either. Hence in general this metric is not recommended.

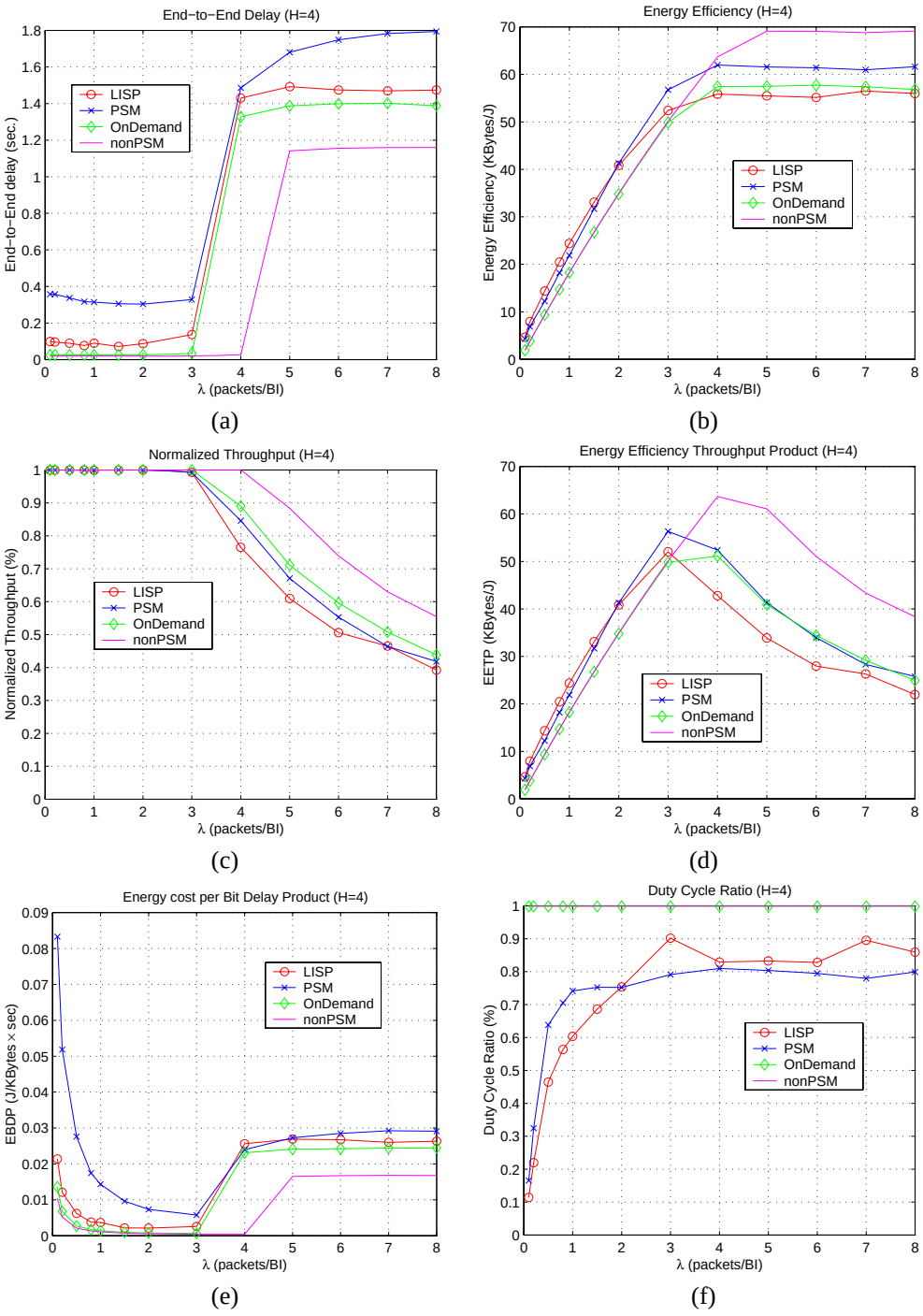


Fig. 8. Simulation I: Effect of traffic load. A tandem topology: $1 \rightarrow 2 \rightarrow \ldots \rightarrow 5$.

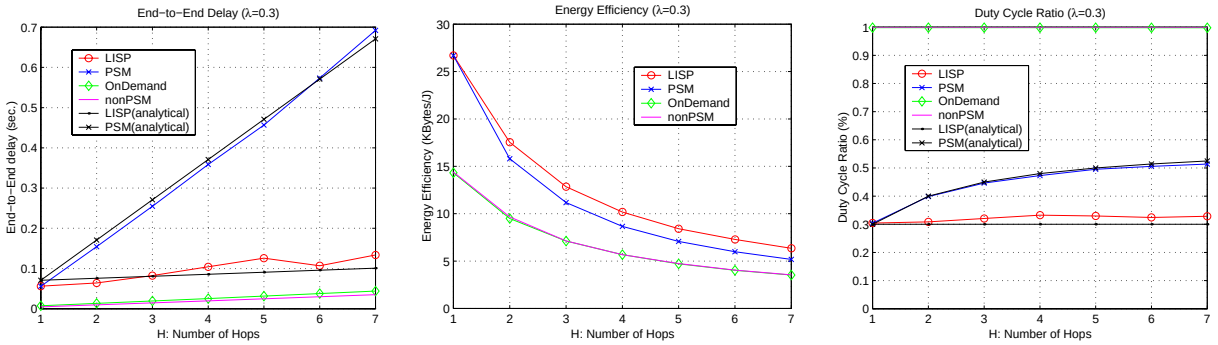


Fig. 9. Simulation II: Effect of number of hops. A tandem topology: $1 \rightarrow 2 \rightarrow ... \rightarrow (H+1)$. $\lambda = 0.3$ packet/BI, or 24Kbps.

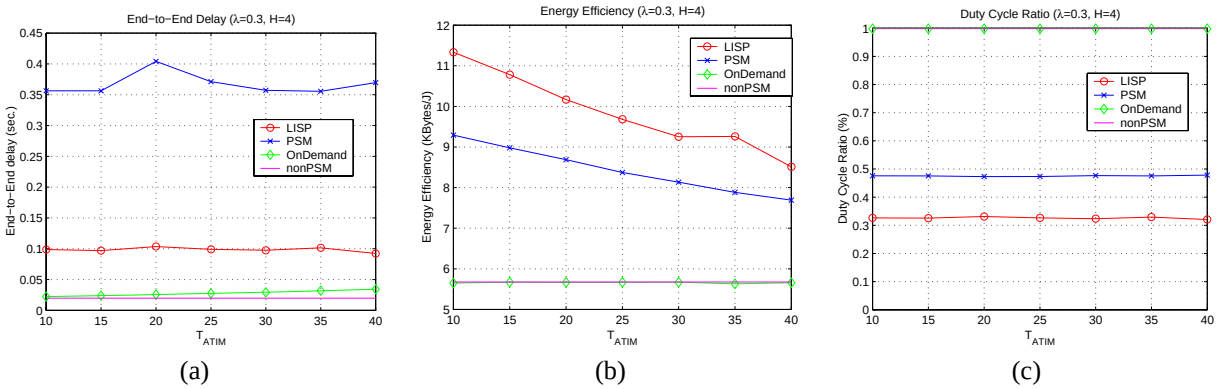


Fig. 10. Simulation III-1: Effect of the ATIM window size. A tandem topology: $1 \rightarrow 2 \rightarrow ... \rightarrow 5$. $\lambda = 0.3$ packet/BI.

- The on-demand approach performs similarly to nonPSM. In on-demand all the nodes en route are kept awake throughout the simulation run once the connection commences. This is because on-demand essentially operates by keeping a node in the active mode for a pre-determined interval upon receipt of packets (upon detection of communications activities). When inter-packet intervals are smaller than the pre-specified timeout values, all the nodes en route will keep awake. In the tandem wireles network, all the nodes are en route, hence the performance of on-demand is almost the same as that of nonPSM. In a sense, on-demand makes its prediction on incoming traffic at a coarse time granularity (i.e., at the connection level). LISP makes the prediction at a finer granularity, and is thus able to profile the traffic at the packet level and utilizes the energy indeed "on demand".

2) Effect of Number of Hops: Fig. 9 illustrates how the performance evolves as the number of hops H increases from 1 to 7 at a relative low traffic load $\lambda = 0.3$packet/BI. The normalized throughput is not depicted since they are all 100% (or close within 0.001% at this traffic load. We also validate the analytical results of end-to-end delay and duty cycle ratio under LISP and PSM in this set of scenarios. The key observation is that , the end-to-end delay of PSM increases approximately linearly as the number of hops increases while that of LISP sticks at around 100ms, one BI. Moreover, LISP achieves the highest energy efficiency as demonstrated in Fig. 9(b). Note that more nodes en route result in a trend of decreasing energy efficiency for all mechanisms.

3) Effect of the ATIM Window Size: The ATIM window size can affect both the network capacity and the energy saving nonnegligibly. Large ATIM window size can accommodate more ATIM-ACK exchanges but provides less available data bandwidth. Thus it may help in scenarios where many interweaving routes are present but the traffic load is not heavy. Large ATIM window size also incurs more energy consuming even without any traffic. The ATIM window size affects the end-to-end delay as well, though may not significantly when the network is not overloaded. It is also of the interest that LISP may be be more sensitive to the ATIM window size because the prediciton chain takes time to finish within the ATIM windows. To address these concerns, we vary the ATIM window size T_{ATIM} in [10, 40] ms and compare the performance. The same tandem topology is assumed: $1 \rightarrow 2 \rightarrow ... \rightarrow (H+1)$.

Fig. 10 and Fig. 11 illustrate the performance with fixed number of hops (H=4) and two levels of traffic load: $\lambda = 0.3$ and 1 packet/BI, respectively. The results confirm our analysis. In the simple lined network, as the ATIM window size increases,

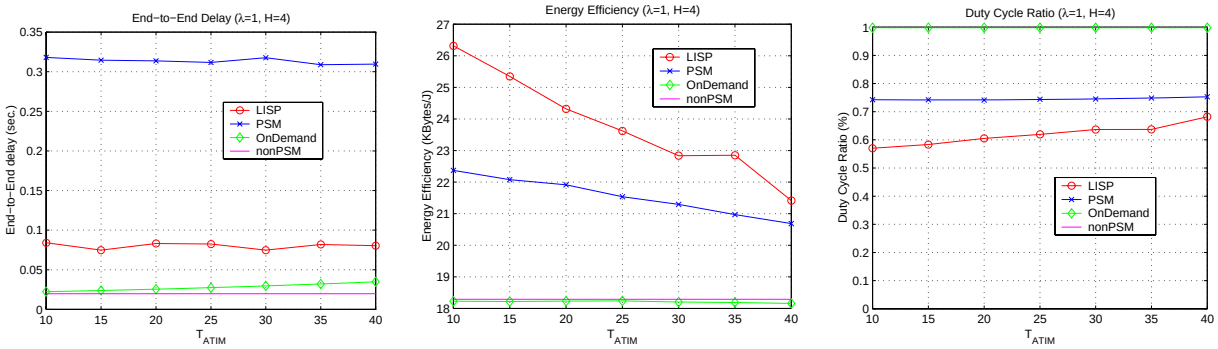


Fig. 11. Simulation III-2: Effect of the ATIM window size. A tandem topology: $1 \rightarrow 2 \rightarrow ... \rightarrow 5$. $\lambda = 1$ packet/BI.

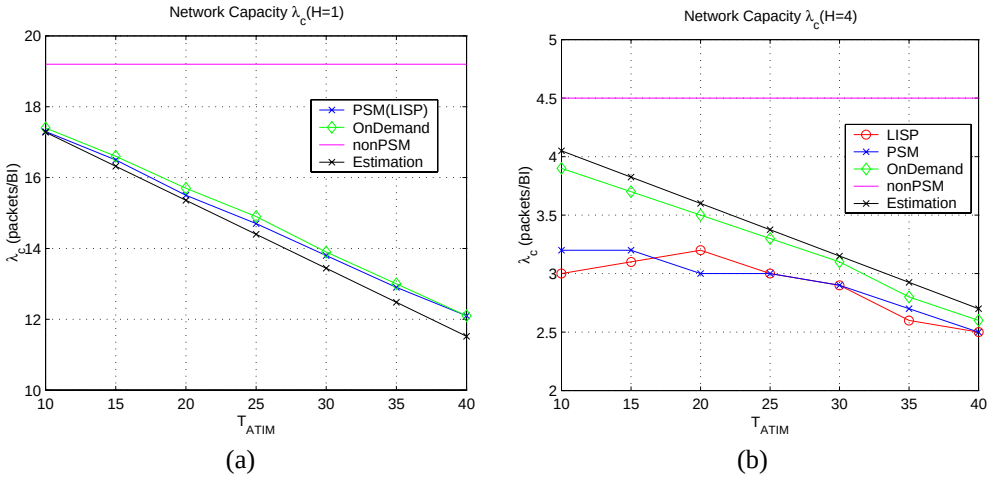


Fig. 12. Simulation IV: Effect of the ATIM window size. A tandem topology: $1 \rightarrow 2 \rightarrow ... \rightarrow H + 1$.

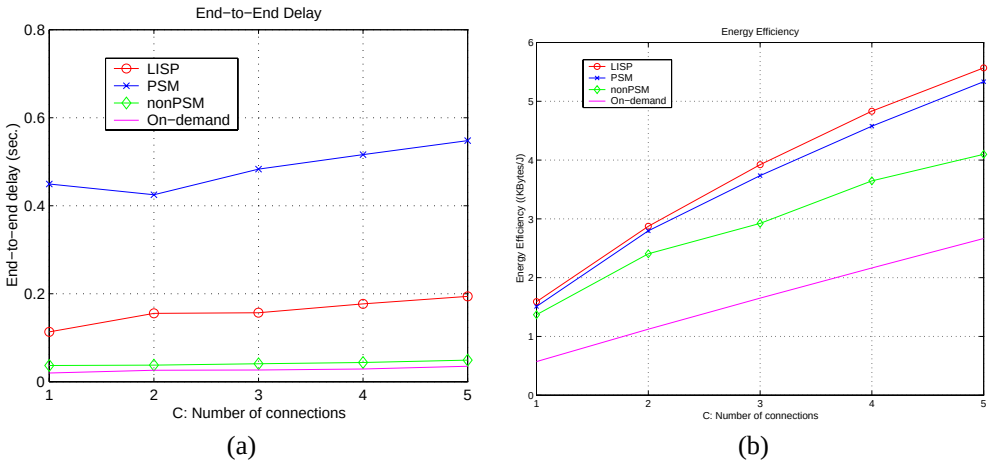
the end-to-end delay of PSM and LISP does not present an obvious trend of monitonical increase or decrease. However, the on-demand approach presents a trend of monitonical increase, though slightly. This is because the ATIM window size does not play two counteracting effects as it does to PSM and LISP. The end-to-end delay increases as the ATIM window size increases simply because nodes have to wait more time to resume forwarding till the ATIM window ends. For all mechanisms, the energy efficiency decreases as more time is devoted to the ATIM window.

Fig. 12 studies the network capacity achieved with increasing ATIM window sizes. We let the traffic rate above which more than 1% packets will be dropped be the network capacity, denoted by λ_c. We get the results at $H = 1$ and $H = 4$ and plot them in Fig. 12 (a) and (b), respectively. The curve marked with "Estimation" is obtained by scaling down the network capacity of nonPSM by a factor of $(1 - T_{ATIM}/BI)$. The results clearly show that the ATIM window size is the major factor that scales down the network capacity of PSM-based mechanisms. This fact is in particular obvious as shown by Fig. 12 (a) when there is a single hop. When there are multiple hops, small ATIM window sizes can result in more failures of ATIM-ACK exchanges and further reduce the capacity even though the available bandwidth is more than that with larger ATIM window sizes. It is observed that the ATIM window size does affect the capacity of LISP more, but only when the size is so small as to affect the completeness of the predication chain (Fig. 12 (b), $T_{ATIM} = 10$).

4) Effect of Packet Size: All the simulation results presented in the above are done with uniform packet size of 1000 bytes. Simulation are also done with packet size of 512 bytes and show the similar trend as the above.

5) Summary: Summarizing the above results, we reach several points:

- The wake-up latency is the dominant factor that increases the end-to-end delay of PSM compared to nonPSM. Under PSM, when the network is not overloaded, the end-to-end delay is at the scale of BI times the number of hops.
- The ATIM window is the dominant factor that scales down the capacity of PSM-based mechanisms.
- LISP can reduce the end-to-end delay of PSM within one BI ideally by predicting traffic at the granuality of packets. By doing so, LISP also achieves higher energy efficiency than PSM under low to moderate traffic load. This gain is lost under



(a) (b)

Fig. 13. Simulation V: 50 nodes are distributed in a plane of 1000m×1000m. C connections are established, each carrying CBR traffic with $\lambda = 0.3$ packet/BI.

heavy traffic load but still remains compared to nonPSM till the network capacity is reached. LISP has a slightly negative effect on the network capacity with very small ATIM window size compared to PSM.

- The on-demand mechanism performs similarly to nonPSM if only nodes en route are considered. As we have discussed, once there is an active flow, the on-demand mechanism tends to keep all the nodes en route awake. Thus it trades off the energy saving of nodes en route for better performance to a large extend, even if the traffic load is light. In comparasion, LISP makes this tradeoff minimal and even no tradeoff at all at a light traffic load.

B. Performance Under a General Scenario

Now we evaluate the performance of LISP and other mechanisms in a more complicated network with multiple connections. In the network to be simulated, 50 nodes are distributed in a plane of 1000m×1000m. C connections are established, each carrying CBR traffic with the rate $\lambda = 0.3$ packet/BI, or 24Kbps, where C ranges from 1 to 5. To ensure each connection traverses multiple hops, we deploy the first 10 nodes, n_{2i} and n_{2i+1}, at position (50, 200+i∗150) and (950, 200+i∗150), respectively, where $i = 0, 1, ..., 4$. The source and destination nodes of the i-th connection are n_{2i} and n_{2i+1}, respectively. The rest 40 nodes are uniformly distributed in the plane. Fig. 13 depicts the simulation results of end-to-end delay and energy efficiency under this scenario. The packet delivery ratios are all 100% or close to within 1%, and are not depicted here. The following observations are made.

- The end-to-end packet delay under LISP is approximately one BI at $C = 1$, and then increases near to 2 BIs as the number of connections increases. This demonstrates the impact of interfering routes and node density. Both factors may result in more failures in building the prediction chain or making a correct prediction. These factors also affect PSM, however. Therefore the overal result is that the end-to-end delay under PSM is 2.8-4.0 as large as that under LISP.
- LISP achieves achieves the highest energy efficiency over all the other compared mechanisms – 6%, 40% and 178% higher than that of PSM, on-demand and nonPSM, respectively. On-demand incurs a higher energy efficiency than nonPSM because because only nodes en route stay awake throughout the duration of connections, and other nodes still benefit from power management. Thus on-demand obtains an energy efficiency between PSM and nonPSM.

VII. Conclusion and Future Work

In this paper, we have proposed a novel extension, LISP, to IEEE 802.11 PSM. For PSM the wake-up latency is the main source that degrades the performance in terms of end-to-end delay. By exploiting the inherent correlation between ATIM_ACK packets and incoming traffic, LISP enables nodes to predict exactly in which BIs packets will arrive. Only in those BIs will nodes switch to the temporary active mode to receive and to forward packets. In this manner, the wake-up latency is reduced and a “freeway” is essentially built on-demand to transport packets without incurring excessive duty cycles. We conduct analytical and comprehensive simulation studies. The impacts of various factors, including traffic load, number of hops, ATIM window size and packet size, are investigated detailedly using a tandem topology. The results confirm that LISP has advantage of significantly less end-to-end delay over PSM and performs much less sensitively to the number of hops; that LISP saves more energy under low and moderate traffic load and consumes less energy all the time compared to nonPSM before the network capacity is reached. In a general scenario simulated, LISP reduces the end-to-end delay of IEEE 802.11 PSM by 65-75%, and increases

the energy efficiency of IEEE 802.11 with and without PSM by 6% and 178%, respectively. The simulation study indicates that the ATIM window is the dominant factor that reduces the network capacity under PSM-based mechanisms compared to that without power management. This suggests a direction for future work.

LISP, however, has its limitations. First, the first packet of each connection is still susceptible to the wake-up latency, because LISP has to go through the learning phase. Second, the performance of LISP degrades under heavy traffic loads and/or presence of multiple connections. This is, in part, due to the fact that LISP indexes the prediction state by links. Without the connection information, a node cannot distinguish traffic indicators for one connection from those for the others. Collisions of pseudo-ACK packets in the case of heavy traffic load also contribute to the performance degradation. All the observations enlighten us to consider several possible research avenues for improving LISP, e.g., how to incorporate more complicated and yet feasible statistical approaches to make traffic prediction, and how to combine LISP with load-balanced routing to achieve better overall performance. These constitute our future work, and will be reported in forecoming manuscripts.

REFERENCES

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

[2] B. Chen, K. Jamieson, H. Balakrishnan and R. Morris, *Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks*, IEEE MobiCom, 2001

[3] L.M. Feeney and M. Nilsson, *Investigating the energy consumption of a Wireless network interface in an ad hoc networking environment*, IEEE InfoCom, April 2001

[4] E.-S. Jung and N.H. Vaidya, *An energy efficient MAC Protocol for wireless LANs*, IEEE InfoCom, June 2002

[5] N. Li, C.-J. Hou and L. Sha, *Design and analysis of an MST-based topology control algorithm*, IEEE InfoCom, April 2003

[6] J. Monks, V. Bharghavan and W.-M. Hwu, *A power controlled multiple access protocol for wireless packet networks*, IEEE Infocom, March 2001

[7] V. Rodoplu and T.H.-Y. Meng, *Minimum energy mobile wireless networks*, IEEE Personal Communications Magazine, pp46-55, April 1999

[8] C. Schugers, V. Tsiatsis, S. Ganeriwal and M.Srivastava, *Topology management for sensor networks: Exploiting latency and density*, IEEE MobiCom, 2002

[9] S. Singh, M. Woo and C. Raghavendra, *Power aware routing in mobile ad hoc networks*, in Proc. of MobiCom, 1998

[10] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, *Power-Saving protocols for IEEE802.11-Based multi-hop ad hoc networks*, IEEE InfoCom, June 2002

[11] H. Woesner, J.-P. Ebert, M. Schlager and A. Wolisz, *Power-saving mechanisms in emerging standards for wireless LANs: The MAC Level Perspective*, IEEE Personal Communications, June 1998

[12] Y. Xu, J. Heidemann and D. Estrin, *Geography-informed energy conservation for ad hoc routing* IEEE MobiCom, July 2001

[13] W. Ye, J. Heidemann and D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies, 2002

[14] R. Zheng and R. Kravets, *On-demand power management for ad hoc networks*, IEEE InfoCom, April 2003

[15] `http://www.pdos.lcs.mit.edu/~benjie/span/span_ns.html`