Micro Power Management of Active 802.11 Interfaces

Jiayang Liu and Lin Zhong
Department of Electrical & Computer Engineering, Rice University
Houston, TX, 77005
[jiayang, lzhong]@rice.edu

ABSTRACT
Wireless interfaces are major power consumers on mobile systems. Considerable research has improved the energy efficiency of elongated idle periods or created more elongated idle periods in wireless interfaces, often requiring cooperation from applications or the network infrastructure. With increasing wireless mobile data, it has become critical to improve the energy efficiency of active wireless interfaces. In this work, we present micro power management (µPM), a solution inspired by the mismatch between the high performance of state-of-the-art 802.11 interfaces and the modest data rate requirements by many popular network applications. µPM enables an 802.11 interface to enter unreachable power-saving modes even between MAC frames, without noticeable impact on the traffic flow. To control data loss, µPM leverages the retransmission mechanism in 802.11 and controls frame delay to adapt to demanded network throughput with minimal cooperation from the access point. Based on a theoretical framework, we employ simulation to systematically investigate an effective and efficient implementation of µPM. We have built a prototype µPM on an open-access wireless hardware platform. Measurements show that more than 30% power reduction for the wireless transceiver can be achieved with µPM for various applications without perceptible quality degradation.

Categories and Subject Descriptors
C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design
C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks

General Terms
Measurement, Performance, Design, Experimentation

Keywords
Energy-efficient wireless, 802.11, power management

1. INTRODUCTION
IEEE 802.11 is one of the most popular high-performance wireless technologies for mobile systems. It provides the infrastructure for not only wireless local-area network and hotspots but also many urban wireless mesh networks. 802.11 interfaces have already become universal on laptops and appeared on high-end mobile phones. Like typical wireless interfaces, they consume significant power, as highlighted in [21][19]. Although 802.11 MAC layer provides a power-saving mechanism, called Power-Saving Mode or PSM [9], it is effective only for elongated idle periods (>100ms).

The focus of this work is on energy reduction in short idle intervals embedded in busy time that are out of the reach of 802.11 PSM. Such idle intervals are abundant because of 1) the gap between high data rate supported by modern 802.11 interfaces and the modest data rates required by many popular applications; and 2) the limitation often set by the wired link, e.g., DSL. Our measurement shows that 802.11 interfaces on laptops can easily spend over 80% of its busy time and energy in waiting for frames through idle intervals below 200ms. Meanwhile, advancement in RF IC design has made it profitable for a wireless interface to enter a power-saving mode for as short as several microseconds (µs), because the time and energy costs of using power-saving modes have been dramatically reduced in recent years [1] [22] [14].

In this work, we present micro power management (µPM) to leverage these opportunities. Our goal is to demonstrate that busy-time power consumption of 802.11 interfaces can be dramatically reduced by judiciously putting the interface into a power-saving mode for idle intervals as short as several microseconds. We address multiple challenges to the practical realization of µPM. First, one may risk losing incoming frames by putting the wireless interface into an unreachable state without cooperation from the infrastructure. µPM addresses this by leveraging the retransmission mechanism of 802.11. With minimal cooperation from the access point, it can save energy while suppressing data loss. Second, by engendering more retransmissions, µPM may introduce frame delay, leading to noticeable changes to upper network layers including applications. To address this, µPM employs history-based prediction to statistically and adaptively contain frame delays. Third, a wireless interface with µPM may impact the throughput of its peers by increasing the workload of the access point. Therefore, µPM regularly assesses the network load and applies power management adaptively.

Driven by a theoretical framework, we provide an efficient implementation of µPM and investigate its strength and weakness through extensive simulations based upon both synthetic and realistic traces. The results show that µPM can reduce busy power consumption for a modestly loaded commercial 802.11 transceiver without sacrificing data loss or noticeable frame delay. We have implemented a prototype of µPM on an open-access wireless research hardware platform. Our measurement-based characterization of the implementation shows that more than 30% power reduction is achieved without noticeable degradation in quality of service.

MobiSys'08, June 17–20, 2008, Breckenridge, Colorado, USA.

In summary, we report μPM, a novel standard compliant power management approach, to effectively reduce power consumption of active 802.11 interfaces in a lightly loaded network with quality of service guarantee. Our work makes the following specific contributions:

- We provide a theoretical framework to evaluate and optimize μPM (Section 4);
- We provide an efficient and effective implementation of μPM. Through extensive simulations with both synthetic and realistic traces, we demonstrate that it can adapt to the traffic and channel condition to conserve energy with very small overhead. We also show that μPM can effectively control data loss with minimal cooperation from the access point (Section 5);
- We demonstrate the practicality of μPM with a prototype implementation on an open-access wireless platform (Section 6).

Aware of the sophistication of 802.11 networking, we do not claim a comprehensive study of the impact of μPM here. Nevertheless, we demonstrate its practicality and effectiveness thus invite more research endeavor to study it further. The rest of the paper is organized as follows. Section 2 introduces related research in wireless energy conservation. Section 3 presents the motivation of μPM and highlights the technical challenges. Section 4 addresses the theoretical components of μPM. Section 5 investigates the efficient and effective implementation of it through comprehensive simulations. It also reveals the strength and weakness of μPM. Section 6 describes our hardware prototype implementation and provides a comprehensive set of measurement-based data to validate the practical implementation of μPM. Section 7 discuses the limitations of μPM and possible solutions as future work. Finally Section 8 concludes the paper.

2. RELATED WORK

To the best of our knowledge, μPM is the first reported work to manage 802.11 wireless interfaces for short intervals (<<100ms) between MAC frames in an application-independent fashion. Many have investigated traffic shaping techniques with cooperation from the application or the network infrastructure [20][8][2]. These techniques employ buffering to accumulate short intervals into a long one for power management, henceforth traffic shaping. They usually introduce frame delays as long as multiple seconds, large enough to impact quality of service for most applications. In contrast, μPM judiciously switches the wireless interface between power-saving and idle modes during short intervals between frames without buffering or cooperation from the application or network infrastructure. The resulted frame delay is invisible to the upper network layers.

Power management in the busy time has also been well supported in wireless technologies intended for highly mobile access, in particular through the time division multiple access scheme. For example, GSM cellular networks employ Discontinuous Transmission (DTX) and Reception (DRX) to power down the radio receiver during active data transfers. 802.16e (WiMax) also provides power-saving modes (sleep and idle modes) that can be applied to busy time.

Unfortunately, 802.11 does not provide similar support for busy time power conservation. Although the standard 802.11 PSM works well for elongated idle time (>100ms), it is not suitable for busy time: the frame latency under 802.11 PSM is too long to satisfy quality of service. A mobile station needs to wait for one beacon interval (usually 100ms) to receive buffered incoming frames. Moreover, the network throughput degrades dramatically when the beacon interval decreases [18]. The Automatic Power Save Delivery (APSD) in 802.11e employs data frames, instead of PS-Poll, to retrieve buffered data from the access point thus can power manage very short idle periods [6]. However, it works well only for traffic with similar data rates in both directions, such as that of VoIP [7]. The authors of [3] proposed to put an active 802.11 interface into a power-saving mode for short periods when it overhears that the access point is communicating with someone else. The effectiveness of the proposed solution relies on peer traffic and is effective in crowded environment with considerable peer contention. Because the interface goes into a power-saving mode only for the time that the access point is communicating with a third party, it does not risk losing incoming frames as μPM. As a result, however, the method is unable to benefit very short intervals, e.g. those in the order of microseconds, because a frame transfer usually takes several miliseconds. In contrast, μPM does not rely on peer traffic and can benefit idle intervals as short as power-saving modes permit, which can be several microseconds. In addition, because μPM puts the interface into a power-saving mode without direct knowledge of the access point's status, it must carefully deal with possible incoming frames. On the other hand, the method in [3] can be considered complementary to μPM: the former works well with an access point busy with its peers, while the latter works well with a lightly loaded access point.

Bounded Slowdown [12] adaptively increases the beacon intervals to reduce the energy overhead of periodic wakeup in 802.11 PSM while containing the TCP packet latency. SPSM [17] schedules the wakeup time to guarantee a desired delay with minimum energy consumption. Wake-on-Wireless [21] and CoolSpot [15] leverage a secondary low-power radio to further reduce the power consumption of elongated idle time in 802.11. All these solutions were targeted at elongated idle periods (>100ms). Their overhead prevents them from benefiting short idle periods as μPM targets at. The rationale behind μPM is that a system component may be power managed for a very short period of time without being noticed. The same rationale has been applied to other system components under very different context. μSleep carefully engineered the operating system so that a mobile device can sleep with display on and wake up upon user input or a timer without being noticed [5]. We employed history record and human psychology to estimate how long it will take a user to respond a mobile device and subsequently manage the device to avoid user-perceptible latencies [25]. μPM, however, does not enjoy the human tolerance of delay: incoming frames may be lost if the wireless interface is unreachable. μPM leverages frame interval prediction and the retransmission mechanisms of 802.11 to contain data loss. In [10], the authors investigated the possibilities of power managing switches in wired local-area networks. They found that ACPI support or extra hardware for buffering frames is necessary to minimize data loss due to power management.

3. MOTIVATION AND BACKGROUND

μPM is motivated by two opportunities described below.

3.1 Network Bandwidth Under-Utilization

While modern 802.11 interfaces usually support the highest possible data rate, many applications only use a small fraction of it. Moreover, bottlenecks in the network infrastructure can also pre-

Table 1: 802.11 interfaces usually see modest data rates and spend significant time and energy in idle periods embedded in busy time (Energy data is based on Intersil PRISM 802.11 transceiver[1])

		Applications						Users			
		IE	IE-Video	Google Talk	Remote Desktop	MSN-Video Call	File Download	1	2	3	4
Short Idle Intervals Out of Busy Time (%)	Time	79.4	77.9	76.3	87.5	73.1	44.5	92.9	93.6	93.6	81.4
	Energy	69.5	67.6	65.5	80.6	61.6	32.2	88.6	89.7	89.7	72.2

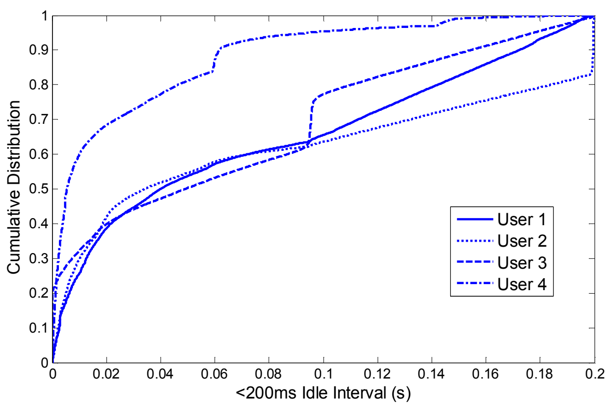


Figure 1: Length distributions of idle intervals below 200ms for 802.11g interfaces from four laptops

vent an interface from achieving its maximum data rate. For example, many home Wi-Fi networks employ DSL connectivity to the Internet with data rate below 1Mbps. Consequently, the wireless interface often completes data transceiving before the next frame is ready, leading to abundant idle intervals during busy time. To highlight this opportunity, we collect network traces for several popular applications and real wireless network usage by four laptop users for a whole week (described in Appendix). We remove idle intervals over 200ms in order to focus on the busy time of the wireless interfaces. Table 1 presents the percentage of time and energy the wireless interface spends in idle waiting out of the total busy time. It shows that the wireless interfaces spend significant time and energy in idle intervals, which presents a great opportunity for energy saving. Such idle intervals are the key bottleneck for 802.11 interfaces to maintain a low energy per bit data transfer when interfaces are not transceiving at the peak data rate. In our recent work [19], we observed that the energy per bit increases fourfold when the 802.11 data rate decreases from 2Mbps to 256Kbps.

3.2 Power-Saving Modes

Today's wireless interfaces provide multiple power-saving modes by disabling different components, therefore with different power savings and wake-up latencies. For example, the PRISM 802.11 transceiver provides four power-saving modes with power reduction from 34% to 90%, with latency from 1us to 5ms, respectively [1]. More importantly, the rapid development in RF circuit design has resulted in power-saving modes of lower power and shorter wake-up latency, as apparent from the 802.11 transceiver and frequency synthesizer design reported in recent years [22] [14]. Due to the energy overhead of mode transition, there is a minimum length of idle interval to justify the use of a power-saving mode. For the PRISM transceiver [1], the minimal length is 1us, 45us, 2.7ms, and 6.4ms for the four power-saving modes, respectively. Figure 1 provides the distribution of the lengths of idle intervals during active data transfers for the real usage traces we collected. Clearly, out of the short idle intervals below 200ms (thus out of the reach of 802.11 PSM), a significant portion are long enough to benefit from power-saving modes supported on modern wireless interfaces.

3.3 Technical Challenges to µPM

µPM aims at leveraging the two opportunities mentioned above to aggressively reduce the power consumption of idle intervals as short as several hundred microseconds. However, there are multiple technical challenges to be addressed.

Firstly, the first challenge is incoming data loss due to putting the wireless interface into a power-saving mode. Fortunately, 802.11 MAC employs a retransmission mechanism for reliability. By leveraging this and carefully timing the unreachable time, µPM can control data loss. With minimal cooperation from the access point, it can completely avoid additional data loss.

The second challenge is frame delay if a frame arrives when the interface is in a power-saving mode. The delay for outgoing frames equals the wakeup latency, which is usually negligible compared to the intrinsic network delay. But the delay for incoming frames can be as long as the unreachable period plus the time to receive the retransmission. To tackle this problem, µPM predicts the next frame arrival time using traffic history to statistically bound delay.

The third challenge is impact on the performance of network peers. By missing transmissions, a wireless interface with µPM will engender retransmissions from the access point thus limit the latter's capability to serve others. Therefore, µPM regularly assesses the network load and applies power management adaptively.

4. THEORETICAL FRAMEWORK

We next address the working principles of µPM and provide a theoretical frame for its analysis and optimization.

4.1 Objective and Constraints

The objective of µPM is to minimize energy consumption while satisfying communication quality constraints. We consider two constraints. The first is to control incoming data loss. This is particularly important for TCP since high data loss rates will frequently restart the TCP congestion window and lead to reduced throughput. The second constraint is to guarantee a certain percentage (p_{const}*•100%) of frames are not delayed by μPM and to bound the maximum delay of those delayed. This is particularly important for UDP traffic because frame delay variation (jitter) is detrimental to streaming media quality rate. The percentage constraint serves as an important knob for controlling communication quality under μPM.

4.2 Overview of μPM

Figure 2 illustrates how μPM works overall using a state-transition diagram. At any given time, μPM has prediction for the arrival time of the next incoming frame and the probabilistic distribution for the arrival time of the outgoing frame. When the wireless interface is in the process of frame exchange for transceiving a data frame, it is in the *Transceive* state. Otherwise, it is in the *Idle* state. The wireless interface will spend the short idle interval between two data frames in the Idle state. μPM essentially divides the Idle state into several sub-states: the *listen* state and multiple *unreachable* states. The listen state is the original idle mode of the wireless interface in which the interface can listen to the traffic, detect incoming frames, and enter the Transceive instantly (Transition 1 in Figure 2).

Upon entering the Idle state (Transitions 2 & 3), μPM determines which sub-state to put the interface into and for how long so that energy is minimized while meeting the constraints based on its idle interval predictions and energy model of the wireless interface. If the sub-state is an unreachable state, μPM also calculates how long it should stay in the listen state afterwards. Consequently, the interface will stay in the listen state for the calculated time after waking up in order to avoid incoming data loss (Transition 4), as will be addressed in Section 4.3. If an outgoing frame arrives when the interface is in an unreachable state, it immediately wakes up and enters the Transceive state (Transition 5). Apparently, if an incoming frame arrives when the interface is in the unreachable state, it will not be received. However, we will soon see in Section 4.3 how the listen period after the unreachable state can be timed to guarantee the reception of the retransmission. When an incoming or outgoing frame arrives during listening, the interface immediately enters the Transceive state (Transition 1). Otherwise, μPM will make new predictions regarding frame arrival times and consider the wireless interface to transit to the Idle state again(Transition 3).

Upon entering the Transceive state (Transitions 1 & 5), μPM updates its predictions of the arrival time for the next frame in the same direction (Transitions 1, 5 & 7). When the interface wakes up from an unreachable state, μPM will calculate its time in the listen sub-state and leaves it there (Transition 4).

μPM aims at reducing power consumption for very short idle intervals and is complementary to 802.11 PSM. When the wireless interface encounters an idle period longer than the listen cycle of μPM, μPM will repeatedly wake up and return to unreachable states. For a synergy with 802.11 PSM, μPM will delegate the power management to the standard PSM when the predicted idle period is significantly longer than 100ms (Transition 6); it can resume control when active traffic is detected again (Transition 7).

Atomic Frame Exchange: To transceive one data frame, 802.11 performs an exchange of one or more control frames (RTS/CTS/ACK) with possible retransmissions. μPM views such frame exchange as atomic, and considers the interface as in the Transceive state until the exchange for one data frame completes. For example, after the reception of RTS, μPM will wait until the interface confirms with an ACK frame to power manage it.

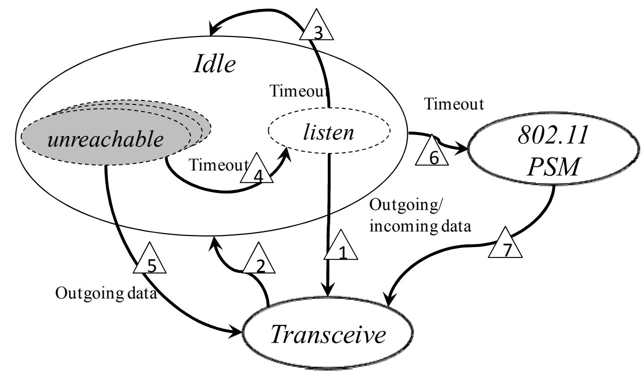


Figure 2: State-transition diagram for μPM, which creates sub-states for the Idle state. Upon entering the Idle state (2 & 3), μPM selects a sub-state

4.3 Incoming Data Loss Control

Turning the interface into an unreachable state recklessly may cause incoming frames loss. To control incoming data loss, μPM takes advantage the 802.11 retransmission mechanism, which is illustrated in Figure 3. Before the access point sends out a data frame, it examines the frame size: if the size exceeds a threshold, it will first perform RTS/CTS handshake with the destination wireless interface and then transmit the data; otherwise, it will transmit the data directly. In either case, it expects to receive a CTS or ACK frame after the initial transmission. For convenience, we call the initial frame sent by the access point the *request frame*, being it the RTS frame if RTS/CTS is employed or the data frame if not; we call the expected CTS or ACK frame after the request frame the *response frame*. If no response frame is received after timeout, the access point will back off briefly to avoid collision before retransmitting the request frame. The back off time is randomly chosen between 0 and the contention window, which doubles each retransmission. The access point will repeat the timeout-retransmit procedure until a response frame is received or the maximum number of (usually seven) retransmissions is reached.

In order to control data loss under realistic conditions, as will be discussed in Section 5, we limit μPM to miss at most the first four retransmissions because in practical 802.11 networks, retransmissions rarely happen more than three times for a frame[26]. Four retransmissions can be as short as four timeouts, assuming a 0 back-off time for each. Hence the wireless interface can stay in an unreachable state for at most about 4ms, or T_{max} (calculated based on 802.11 specifications [9]). If the wireless interface is unreachable for T ($T < T_{max}$), it must listen for long enough to catch the next possible retransmission. The listen time is calculated from the maximal number of possible retransmission misses determined by T. We use $\tau_{awake}(T)$ to denote the minimum listen time after an unreachable time of T. For example, the listen time to catch the fifth retransmission must be as long as one timeout plus a back-off with the full length of the contention window, which is about 12ms (calculated based on 802.11 specifications [9]). Such maximum unreachable time and minimum listen time obviously set limits to the power-saving capability of μPM, which will be explored in

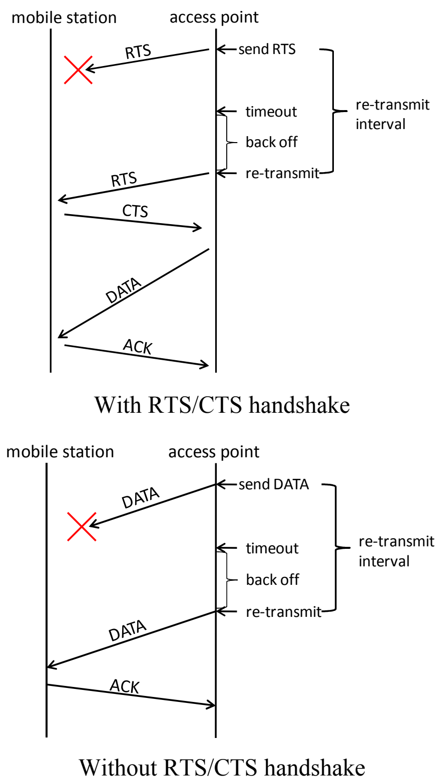


Figure 3: Retransmission mechanism of 802.11

Section 5. On the other hand, they also confine the maximum frame delays to about 16ms, which is quite modest in view of the intrinsic network delay.

4.4 Incoming Frame Delay Guarantee

In this section, we build the theoretical foundation for µPM to guarantee the frame delay constraint and optimize the tradeoff between frame delay and energy saving. It will drive the efficient implementation of µPM to be discussed in Section 5.

Since the wireless interface transits to the Transceive state immediately upon an outgoing frame, the delay is below 1ms therefore negligible. However, it is extremely important to guarantee a certain percentage of frames are not delayed due to µPM, because 1) missed incoming frames will be retransmitted and therefore waste the network throughput, and 2) missing the initial transmission increases the probability of data loss when the channel is not ideal. Therefore, our implementation of µPM employs an adaptive frame delay constraint: when network demand increases or the channel condition degrades, it tightens the constraint; and vice versa (described in Section 5.3). Frame delay constraint serves as an important knob for controlling communication quality under µPM.

4.4.1 Problem Formulation

µPM guarantees at least p_{const}•100% of the incoming frames are not delayed by predicting the frame arrival time based on history. Assume the frame arrival is a stationary random process. Let T_{idle} denote the length of the idle interval, which is a random variable; its cumulative distribution function is $F_{idle}(t) = \Pr\,(T_{idle} \leq t)$.

Our prediction algorithm estimates the next idle interval by finding T_{est} such that $\Pr(T_{est} \leq T_{idle}) = P_{thresh}$. When $T_{est} \leq T_{idle}$, no frame arrives during the estimated idle interval. So if the interface stays in an unreachable state for T_{est} and then wakes up, the probability of having no frame delay is P_{thresh}. P_{thresh}, called *threshold probability*, is therefore critical in guaranteeing the frame delay constraint. Knowing P_{thresh}, the prediction algorithm estimates the next idle interval as $T_{est} = F_{idle}^{-1}(1 - P_{thresh})$, because

$$\Pr(T_{est} < T_{idle}) = 1 - \Pr(T_{idle} \leq T_{est}) = 1 - F_{idle}(T_{est})$$
$$= P_{thresh}$$

where $F_{idle}^{-1}(p)$ is the reverse function of $F_{idle}(t)$.

A simplistic solution would employ a static $P_{thresh} = p_{const}$. However, as we will soon see, this may not be optimal. Therefore, µPM considers P_{thresh} as a random variable with probability density $f_{thresh}(p)$. To guarantee the delay constraint, we must have:

$$\mathrm{Exp}\{P_{thresh}\} \geq p_{const} \tag{1}$$

Therefore, the core of the prediction algorithm is to find $f_{thresh}(p)$ such that $\mathrm{Exp}\{P_{thresh}\} \geq p_{const}$ with the maximal energy saving.

To simplify the analysis, we only consider one unreachable state for now. Then the energy saving ΔE is a linear function of T_{est}:

$$\Delta E = \Delta P \cdot T_{est} - E_{overhead} \tag{2}$$

where ΔP is the power saving in the unreachable state, and $E_{overhead}$ is the energy overhead to transit between the unreachable and full power states. Note that Equation (2) holds no matter whether T_{est} is greater than T_{idle} or not.

So our goal is to maximize the expectation of ΔE:

$$\mathrm{Exp}\{\Delta E\} = \Delta P \cdot \mathrm{Exp}\{T_{est}\} - E_{overhead} \tag{3}$$

It reduces to maximizing the expectation of T_{est}:

$$\mathrm{Exp}\{T_{est}\} = \mathrm{Exp}\{F_{idle}^{-1}(1 - P_{thresh})\} \tag{4}$$

Now we can see a static P_{thresh} can be inferior in some situations. If $F_{idle}^{-1}(p)$ is convex, as is true for many frame arrival process models [16], we can apply Jensen's Inequality [11]:

$$\mathrm{Exp}\{T_{est}\} = \mathrm{Exp}\{F_{idle}^{-1}(1 - P_{thresh})\} \geq$$
$$F_{idle}^{-1}(\mathrm{Exp}\{1 - P_{thresh}\}) = F_{idle}^{-1}(1 - p_{const}) \tag{5}$$

Because Jensen's Inequality provides that the minimum energy saving can be achieved when $P_{thresh} = p_{const}$, a static P_{thresh} will indeed save least energy in this case. This is why we choose to make P_{thresh} dynamic for maximum energy savings no matter whether $F_{idle}^{-1}(p)$ is convex or not.

4.4.2 Discrete Solution

To maximize (4), however, $F_{idle}^{-1}(p)$ is not usually known in a close form and can be only estimated numerically from the history. For a practical and scalable solution, we discretize the value of P_{thresh}. and solve the optimization problem by linear programming. and Assuming there are n levels of threshold probabilities, $P_{thresh} \in \{P_1, P_2, \ldots, P_n\}$, we calculate the expectation of P_{thresh} as

$$\mathrm{Exp}\{P_{thresh}\} = \sum_{i=1}^{n} P_i \cdot Y_i \geq p_{const} \tag{6}$$

where $Y_i = \Pr\,(P_{thresh} = P_i)$. And the optimization objective reduces to

$$\text{Exp}\{T_{est}\} = \sum_{i=1}^{n} F_{idle}^{-1}(1-P_i) \cdot Y_i \qquad (7)$$

So the problem reduces to find Y_i that maximize (7) under the constraint of (6).

With P_i pre-defined and $F_{idle}^{-1}(1-P_i)$ estimated from traffic history, we can solve the problem with linear programming. It is important to note that for stationary traffic, the problem only needs to be solved once for all frames. In Section 5, we will further show that the solution can be significantly simplified so that its computational load is negligible and the simplified solution can be applied more often to address non-stationary traffic.

Multiple Unreachable States: µPM considers multiple unreachable states using the maximum energy saving $\Delta E_m(T_{est})$ given estimated idle interval T_{est}. $\Delta E_m(T_{est})$ is a pre-known but non-linear function because the more power-saving a state, the higher state transition overhead. Given $\Delta E_m(T_{est})$, we need to maximize

$$\text{Exp}\{\Delta E_m(T_{est})\} = \sum_{i=1}^{n} \Delta E_m(F_{idle}^{-1}(1-P_i)) \cdot Y_i \qquad (8)$$

under the constraint of (6). The optimization problem can be solved with linear programming as well.

Incoming Frame Arrival Time Estimation: When the interface enters an unreachable state, incoming frames may be delayed. We estimate its original delay as follows. When the interface receives a request frame, we can examine its frame header to see whether it is retransmitted. If so, we assume the delay due to retransmission is uniformly distributed between 0 and the last unreachable time and subtract it from the measured idle interval in order to keep an accurate statistic of idle intervals.

Domain of P_{thresh}: The domain of P_{thresh} is [0, 1] in the optimization problems formulated above. Because of the limit of maximum sleep time as described in Section 4.3, however, the history records larger than T_{max} will be cut off to T_{max} so that the maximum sleep time is less than or equals T_{max}. Therefore, if P_{limit} is the percentage of history records larger than T_{max}, the lower bound of P_{thresh} should be P_{limit}: ratio of non-delayed frame lower than that will not be achieved based on the history. Meanwhile, the upper bound of P_{thresh} is 1 since we can always achieve that by not sleeping. We will discuss how to discretize P_{thresh} further in Section 5.2.1.

4.4.3 Avoid Consecutive Frame Delays

Although the delay introduced by µPM to incoming frames is in the order of milliseconds and usually trivial compared to intrinsic network delay, it may harm the network performance significantly when the incoming data rate is high so that the frame interval is shorter than several milliseconds. In this case, multiple frames may be queued at the access point when the mobile interface is in an unreachable state. In addition, these frames will not be sent out continuously when the interface is awake, but separated by a random back-off time period. So if the interface goes back to sleep immediately after receiving one frame, the following queued frames may be also delayed. To avoid delaying multiple consecutive frames, µPM makes the interface stay in listen state for a short period of time which is proportional to the current traffic data rate. This method guarantees that delay of a frame will not produce an avalanche.

Table 2: Data for Intersil PRISM [1]

Mode	Power (mW)	Energy overhead (µJoule)	Latency (µs)	Profitable idle time (µs)
Idle	947	0	0	>0
PS-1	627	~0	1	>1
PS-2	231	14	25	>45

4.5 Unreachable and Listen Timing

Upon entering the Idle state (Transitions 2 & 3 in Figure 2), µPM calculates the unreachable time, $T_{unrchble}$, as the smaller of T_{est}, the predicted idle interval, and T_{max}, the maximally allowable unreachable time discussed in Section 4.3 (4ms in our implementation). It then selects the most energy-efficient sub-state of the Idle state for the wireless interface. If the chosen sub-state is an unreachable state, the wireless interface return to the listen sub-state after $T_{unrchble}$ and will stay there for $\tau_{awake}(T_{unrchble})$, as discussed in Section 4.3, in order to catch the next possible retransmission if it missed a transmission during the unreachable time. When the listen period expires without receiving a new frame, µPM updates the history with the observed idle interval so far and transits to the Idle state, making new prediction for the idle interval and selecting a sub-state again (Transition 3).

4.6 Sub-State Selection for Idle State

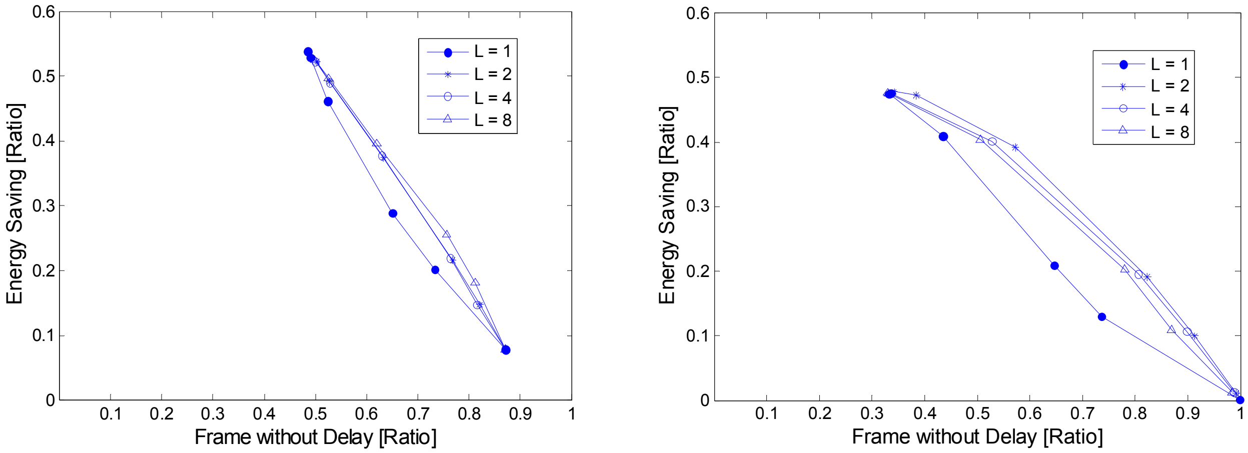
With the unreachable time $T_{unrchble}$, µPM selects a sub-state for the Idle state in order to consume the least energy. However, it has to consider outgoing traffic because the wireless interface wakes up immediately upon an outgoing frame: ignoring it may lead to unnecessary mode transitions and net increase in energy consumption. Therefore, µPM employs the probability density of idle intervals for outgoing frames to check whether the sub-state selected will incur less energy than the idle sub-state under only the outgoing traffic. If so, the sub-state will be accepted; otherwise, a less power-saving sub-state will be tried.

5. SIMULATION-DRIVEN REFINEMENTS AND EVALUATION

In this section, we investigate the efficient and effective implementations of µPM and thoroughly evaluate them with simulation based upon both synthetic and realistic wireless frame traces.

5.1 Two Complementary Simulators

We build two simulators with complementary strengths to evaluate the practical implementations of µPM. The MATLAB simulator reads in pre-collected frame traces (described in Appendix) and emulates the behavior of a wireless interface with µPM applied in different ways. If a frame arrives when the network interface is in unreachable state, the simulator regards it as a delayed frame. While exposing µPM to the dynamics of realistic traces, the MATLAB simulator is limited in that it only applies static traffics. Hence it cannot reveal the interaction between the wireless interface and the rest of the network, including retransmission and data loss. The NS-2 based simulator complements the MATLAB simulator. It emulates an access point with any number of mobile stations of which any can employ µPM. It allows us to apply controlled network conditions to any number of mobile stations with µPM. We employ Poisson traffic of different rates for this purpose.



(a) MATLAB (b) NS-2

Figure 4: Tradeoffs between energy saving and frame delay for µPM with different numbers of levels of threshold probabilities

While Poisson process may not be an accurate model for all network traffic, we consider it a tool complementary to the reality. It is worth noting that the all frames in our NS-2 simulation are below the RTS/CTS threshold thus RTS/CTS is not employed. As a result, the entire data frame will be retransmitted once lost. This leads to higher overhead for retransmission and a higher throughput impact by µPM, if any.

Our simulations employ the data for Intersil PRISM interface because its complete profile is available [1], described in Table 2. However, it is a relatively old implementation. Therefore, the NS-2 simulator assumes 11Mbps as the maximum data rate. However, since our traces were collected from 54Mbps interfaces, the MATLAB simulator still uses 54Mbps. Moreover, the state-of-the-art implementations, including MAX2829 used in our experiments in Section 6 [13], have power-saving modes with more percentage power reduction with smaller overhead [22] [14]. Therefore, we expect µPM will perform even better with modern 802.11 interfaces than what reported below.

We use the MATLAB simulator to evaluate the frame interval prediction of µPM, leveraging the dynamics from realistic network traces; we use the NS-2 based simulator to evaluate the dynamic interaction between a wireless interface and its access point, in particular for frame delay and loss. Both report the ratio of delayed frames. However, only the NS-2 based simulator produces accurate frame delays and data loss. They also calculate the ratio of energy reduction in idle intervals during active data transfer (<200ms), as described below.

5.1.1 Energy Saving Ratio Calculation

In this section, we employ the energy saving ratio out of the idle intervals (<200ms) to gauge the energy-saving capability of µPM. It is calculated as follows. We examine the traces produced by each simulation and calculate the energy consumption in all idle intervals below 200ms and compare it to that if unreachable states were not applied in order to generate the energy saving ratio.

It is important to note that such energy saving ratio is not exactly the same as energy saving ratio per bit data transfer, which would be the ultimate gauge for energy efficiency of active wireless interfaces. However, we choose it over energy saving ratio per bit to present the evaluations of µPM due to the following reasons. First,

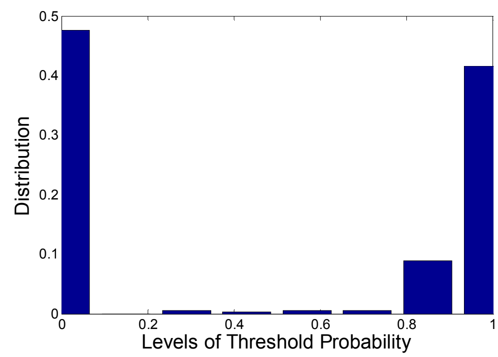
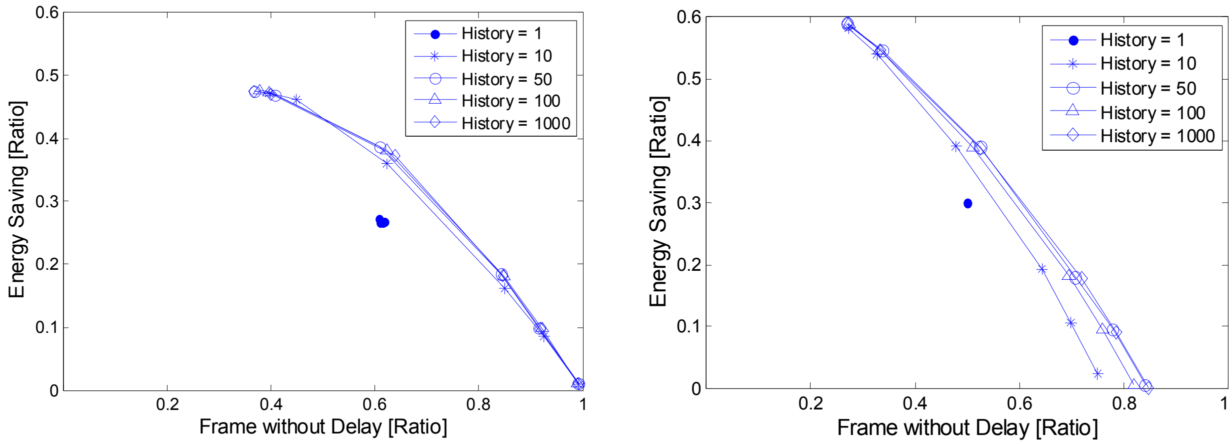


Figure 5: Distributions of the level assignments when eight levels are employed

as we will see later in this section, µPM adaptively guarantee almost the same throughput as that when it is not applied. Second, although µPM may increase the idle time of the wireless interface and incur more transmissions in the access point, it does not incur extra transmissions in the applied wireless interface. Therefore, the energy saving ratio used here is very close to energy saving ratio per bit. Third and most importantly, energy saving ratio of idle intervals better exposes the behavior of µPM. For example, when it is zero, we know µPM does not apply any unreachable states at all and therefore should expect it to behave as it were not applied. This is critical to reveal the adaptation of µPM. It complements throughput to offer a complete picture regarding the performance. In contrast, energy saving ratio per bit would mix the impacts of µPM on idle interval energy and throughput.

5.2 Frame Interval Prediction

Frame interval prediction is a critical component of µPM to guarantee frame delay constraint and minimize its impact on network throughput. The theoretical solution described in Section 4 assumes a stationary traffic and can be potentially compute-intensive. In the following, we investigate its efficient and effective implementation. Our experimental results clearly show that it can be implemented efficiently with minimal computational load, through a history as small as 10 and just two dynamic threshold probabilities.



(a) NS-2 (b) MATLAB

Figure 6: Tradeoffs between energy saving and frame delay constraint for μPM with different history sizes

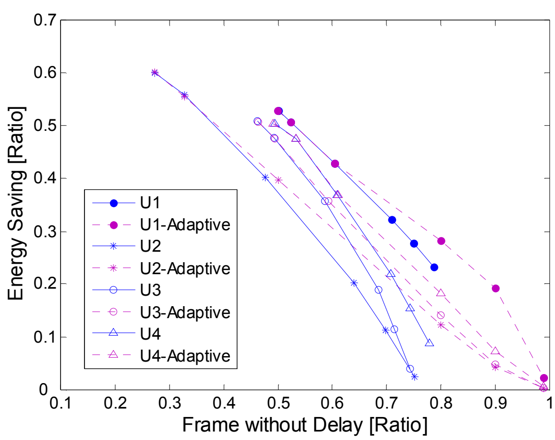


Figure 7: MATLAB with traces from four users (U). The adaptive scheme helps guarantee the delay constraint (up to 0.99) and improves energy saving under the same delay constraint over static μPM

5.2.1 Levels of Threshold Probabilities

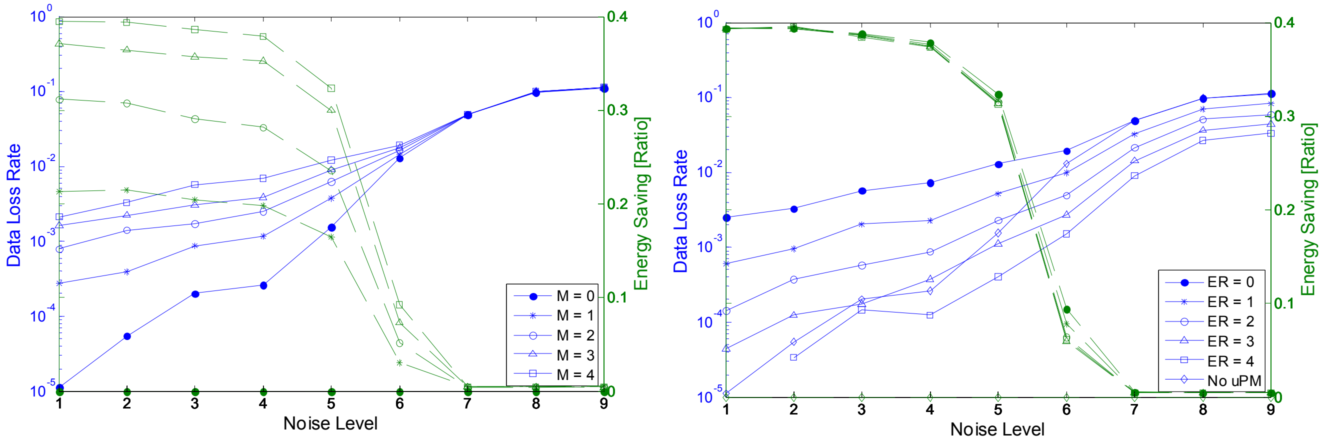
Figure 4 presents the results based on both NS-2 and MATLAB simulations. All implementations of μPM employ a history of the recent 100 frame intervals. We vary the frame delay constraint from 0.1 to 0.99 to produce a tradeoff curve for energy saving ratio and delayed frame ratio. Different implementations of μPM employ different numbers of levels of dynamic threshold probabilities, from 1 (static) to 8, with the levels evenly distributed between 0.01 and 0.99. In the NS-2 simulation, there is only incoming Poisson traffic from the access point. The figure shows clear improvement from a static threshold probability (L=1) to dynamic ones (L=2,4,8), which corroborates our theoretical analysis in Section 4.3. More importantly, it shows that using two levels of threshold probabilities are as good as using more levels, which indicates that the prediction algorithm presented in Section can be dramatically simplified. For example, linear programming will not be necessary.

An examination of the level assignment during simulation highlights the reason for the effectiveness of just two levels. Figure 5 presents the distribution of level assignments for the MATLAB simulation when eight levels of threshold probabilities are employed. It clearly shows that the two extreme levels (0.01 and 0.99) account for most (~90%) of the assignments. Therefore, we will employ only two levels of threshold probabilities in the evaluations below.

5.2.2 History Size and Update

μPM maintains a history of recent idle intervals for both incoming and outgoing frames, respectively, for idle interval prediction. It updates the history record upon entering the Transceive state (Transitions 1, 5 & 7 in Figure 2). If it does not receive any frame in the listen state, it also updates the most recent history record with the idle time since last transceiving. In Section 4.2, we assume the frame arrival is a stationary process. Therefore, the larger the history size, the better to obtain an accurate distribution of the frame intervals. However, a large history increases the computational load significantly and makes μPM irresponsive to the dynamics of traffic when it is non-stationary. To explore the influence of history size, we conducted NS-2 simulations with various history sizes and frame delay constraints from 0.1 to 0.99 when a μPM managed interface talks to the access point in downlink Poisson traffic. Our simulation results in Figure 6 indicate that history sizes over 10 deliver only marginal improvement. Therefore, our implementation of μPM only employs 10 recent intervals as history. Such a short history of frame intervals, combined with only two levels of threshold probabilities, simplifies the frame interval prediction algorithm significantly.

Figure 6 (a) also shows that the frame delay constraints are perfectly satisfied with stationary traffic. This confirms the effectiveness of our theoretical analysis presented in Section 4.4. On the other hand, the short history limits the frame delay constraints for dynamic traffic. We next show how adaptation of the simple algorithms can be employed to address this.



(a) (b)

Figure 8: (a) Allowing missing more retransmissions (M) increases the energy saving at the cost of more data loss; (b) Extra retransmissions (ER) for unreachable nodes suppress data loss

5.3 Adaptation

The idle interval prediction and the short history size allows µPM to adapt to its own traffic and channel condition, as will be demonstrated in Sections 5.4 and 5.6. We next provide adaptive solutions for µPM to guarantee frame delay constraint and to minimize negative impact on peers.

Adaptive Unreachable for Frame Delay: As apparent from Figure 6, µPM yields different delayed frame ratio from the constraint. The reason is multiple. First, the frame intervals can be non-stationary under realistic settings, violating the assumption made in Section 4. Second and importantly, the short history makes it difficult to obtain a fine-grained statistics of the frame intervals. Therefore we devise a simple scheme for µPM to adaptively guarantee the delay constraint, described below.

> *When µPM finds it is profitable for the wireless interface to enter an unreachable state, it may choose not to do so base on a probability, which we call sleep probability and is initialized as 1. We calculate the delayed frame ratio as the weighted moving average of the delay history: a frame contributes 0 if it is not delayed, 1 otherwise. If the ratio is below the constraint, the sleep probability decreases (by 10%), and vice versa.*

Figure 7 highlights the effectiveness of the proposed adaptive scheme in guaranteeing the tight targeted delay constraints. It also improves energy saving given the same delay constraint. Henceforth, we use this adaptive scheme in all experiments below.

Adaptation to Network Load: µPM engenders retransmissions of incoming frames. Because an 802.11 access point will not transmit to other nodes until it finishes the frame exchange for the current outgoing frame, extra retransmissions can degrade the network throughput. Therefore, µPM must apply unreachable states with respect to the load in the rest of the network. We devise an adaptive scheme as follows.

> *When the wireless interface is in the listen state, it overhears traffic between the access point and its peers. µPM employs this opportunity to assess the traffic by accounting the throughput, a, from the access point to its peers. It estimates and tracks the load to the access point, Γ, with* $\Gamma_{current} = 0.75 \cdot \Gamma_{previous} + 0.25 \cdot a$, *which leads to a fine-grained yet robust assessment. µPM increases the frame delay constraint (by 20%) when the load is over 40% of the access point capacity and vice versa.*

The values of the parameters used above are heuristic. Our simulations show that their exact values do not matter much for the effectiveness of µPM. However, we conjecture that it may be possible to obtain optimal values based on knowledge of peer traffic and channel conditions, which is out the scope of this paper. It is also important to note that the load assessment is carried out without incurring extra RF activities in the wireless interface.

5.4 Data Loss Control

So far we have assumed an ideal channel that there is no intrinsic frame loss. With this assumption, µPM guarantees zero data loss by catching at least one re-transmission of the frame. However, missing retransmissions essentially increases the probability of data loss under non-ideal channels. Assuming the intrinsic probability for a frame to be lost is P, missing a retransmission will increase the data loss rates by $1/P$ times. Since our frame interval prediction enables µPM to catch the first transmission of p_{const} of the incoming frames, we can estimate the data loss rate will be increased by at most about $(1 - p_{const})/P^n$ if µPM will miss at most *n* transmissions of the same frame. Note this is a very pessimistic estimation. Without change to 802.11, this is unfortunately the hard limit to µPM. Figure 8 shows how data loss rates increase as the channel noise level increases for µPM implementations set to miss one to four retransmissions. Since data loss involves the interaction between the wireless interface and the access point, we employ the NS-2 based simulator with 1Mbps downlink Poisson traffic in this section and set $p_{const} = 0.5$ with adaptive frame delay control.

5.4.1 Cooperative Access Point

We propose a simple modification to the access point to dramatically suppress the negative impact on data loss, which we call cooperative access point. When µPM decides to put a wireless interface into an unreachable state, it employs an unused bit in the control field of the last outgoing frame before unreachable to notify the access point regarding its decision. The notification bit in the

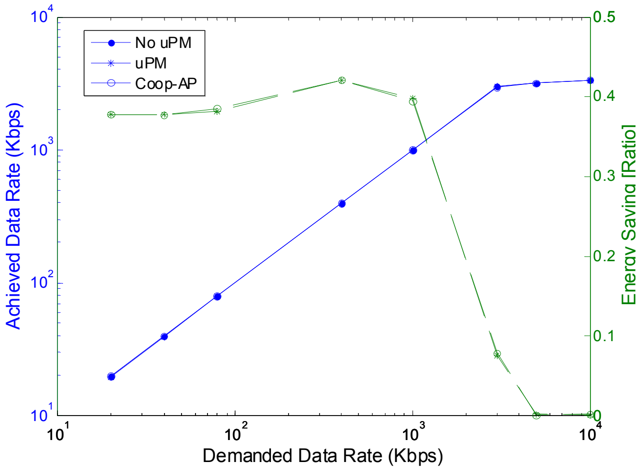


Figure 9: Achieved data rates and energy saving under various demands

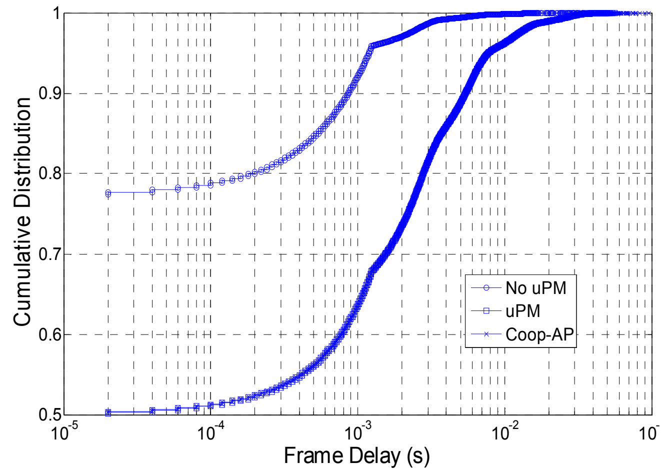


Figure 10: Frame delay distributions. µPM and Coop-AP only increases delays of multiple ms significantly

control field can be the "More Data" bit, which is intended for the access point to notify the wireless interface in 802.11 PSM that the latter has more data to receive. Upon detecting the bit is set, a cooperative access point will attempt more retransmissions for the next frame destined to the corresponding node if necessary, to make up possible missed transmissions due to µPM. Figure 8(b) shows how extra retransmissions improve the data loss of the µPM that will miss at most four transmissions. Three extra retransmissions can guarantee a better data loss rate than that without µPM. For the extra retransmissions, we reset the contention window. Figure 8(b) also shows the cooperative access point achieves the about same energy savings as µPM without it.

5.5 Summary of µPM Implementation

Our evaluations have led to an extremely simple yet effective implementation of µPM, which can be efficiently realized in the baseband processor of an 802.11 interface with negligible overhead. It only keeps the most recent 10 idle intervals and employs only two levels of threshold probabilities. Therefore, instead of sorting and linear programming, µPM only needs to choose between the minimal and the maximal from the 10 history record as the prediction based on a probability obtained by solving a single linear equation reduced from Equation (8).

Based on the predictions, µPM calculate the unreachable time $T_{unrchble}$, as described in Section 4.5. It then identifies the most profitable power-saving mode as described in Section 4.6. If the power-saving mode is unreachable, it decides whether to put the wireless interface into it according to the awake probability, which is adapted upon the reception of an incoming frame. In all evaluation below, we only allow the miss of maximal four retransmissions and set the default delay constraint as 0.5.

µPM then marks the notification bit based its decision in the header of the last outgoing frame (either a DATA or ACK frame) to notify the access point. It finally puts the interface into the unreachable state upon finishing the frame exchange for the current data frame. However, the access point may or may not cooperate as described in Section 5.4. In our evaluation below, a cooperative access point will have at most three extra retransmissions. We use µPM and Coop-AP to denote µPM without and with a cooperative access point, respectively.

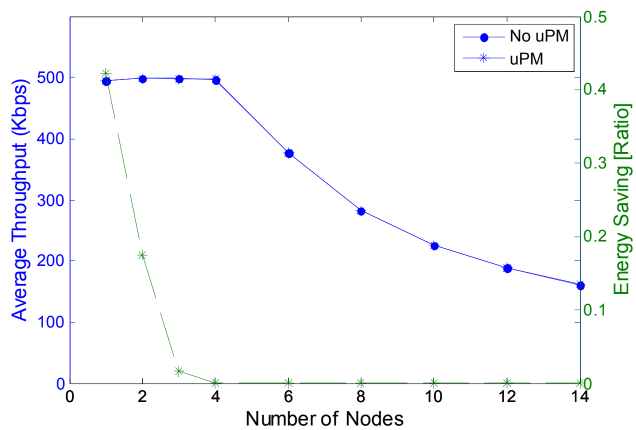


Figure 11: Impact on peer throughput due to µPM

According to Figure 9, µPM can easily reduce energy consumption in idle intervals by 30% for the Intersil PRISM 802.11 transceiver. This will lead to about 20% busy-time energy reduction according to Table 1. As aforementioned, µPM will be able to conserve even more energy with state-of-the-art 802.1 interfaces, which have power-saving modes with more percentage power reduction and lower overhead. In addition, as wireless transceivers dominate the overhead of mode transitions due to the frequency synthesizer, µPM can readily power manage most other components in 802.11 interfaces, including the baseband processor, thus achieving even greater energy reduction in active 802.11 interfaces [1].

5.6 Impact on Network Performance

In this section, we evaluate the impact of µPM on network performance in terms of throughput and frame delays.

Self Performance: The idle interval prediction allows µPM to adapt to channel condition and its own data rate. In Section 5.4, we showed that µPM adapts to channel condition. When the noise level increases, the wireless interface will spend an increasing percentage of time in re-transceiving corrupted frames, leading to shorter intervals, and µPM will adapt accordingly. Figure 8 shows that µPM stops applying power-saving modes (zero energy saving ratio) and approaches the data loss rate without µPM. Figure 9 shows that a wireless interface under µPM will deliver almost the same data rates as demanded by the access point. The energy saving achieved by µPM first increases as data rates increase and eventually decreases due to shrinking idle intervals. The reason that energy saving increases first is because that when the data rate is low, µPM is limited by the maximum unreachable time set by the retransmissions (T_{max}), instead of the predicted length of idle intervals, as discussed in Section 4.5.

Figure 10 presents the frame delay distributions for a wireless interface receiving 1Mbps Poisson traffic with no µPM, µPM, and Coop-AP, respectively, with a frame delay constraint of 0.5. It shows that µPM and Coop-AP impact frame delay distribution only with significantly more frame delays of multiple milliseconds, which is negligible to most mobile wireless access applications. The percentage of delays over 50ms is about the same for all three cases.

Peer Throughput: µPM may also impact the throughput of peers served by its access point with more retransmissions in downlink. We use the NS-2 simulator to emulate an infrastructure network of one access point and one to fourteen mobile nodes uniformly distributed on a circle of 50-meter radius with the access point being the center. Each node has independent, continuous downlink Poisson traffic at 500Kbps. Figure 11 presents the average downlink throughput when µPM and Coop-AP are applied to all the mobile nodes in the network, respectively. It shows that the achieved throughput is very close to that without µPM. It highlights the effectiveness of µPM in adapt to the load imposed on the access point by peers using the simple method described in Section 5.3.

A close look at Figure 11 reveals that the energy saving ratio in the nodes rapidly decreases to zero as the number of nodes increases. This is because each node in the simulation has a continuous traffic from the access point. In reality, however, wireless traffic is likely to be burst and intermittent. Therefore, even with a large number of peers, the wireless interface with µPM may still be able to conserve energy as µPM can rapidly adapt to the load of the access point.

Initially, we suspected that µPM might aggravate the hidden node problem [9] by putting the wireless interface into unreachable states. However, our simulation results with a large number of mobile nodes with uplink traffic show that this is no difference between µPM and no µPM, indicating that µPM avoids the problem by limiting the length and duty cycle of unreachable time.

6. PROTOTYPE REALIZATION OF µPM

We have realized a prototype of µPM based on the Rice WARP platform [23], an open source platform for wireless communication system research.

6.1 Experiment Setup

We use the WARP platform as both the 802.11 access point and the interface with µPM. WARP employs a Xilinx FPGA with a PowerPC core for baseband processing and a RF daughter board, as shown in Figure 12. The RF board consists of interfaces with FPGA, a MAX2829 802.11 transceiver [13], RF frontend, and clock inputs. The programmability of the PowerPC core allows us to implement µPM.

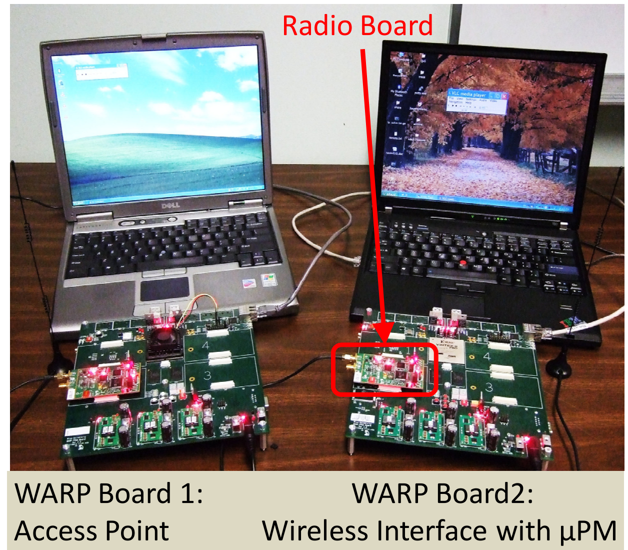


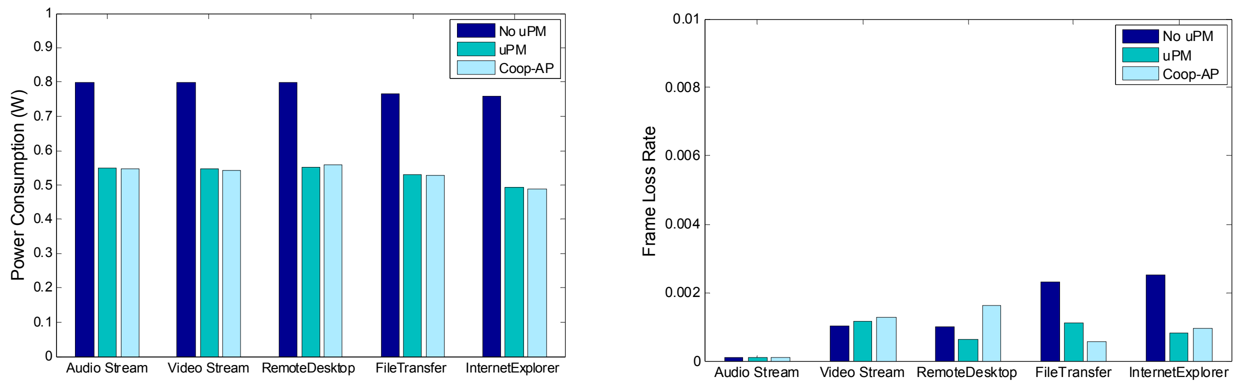
Figure 12: Two WARP boards as the 802.11 interfaces to implement µPM

We employ two identical WARP boards, which work at channel 11 of 2.4GHz band with throughput up to 2Mbps, due to limitations set by current WARP implementation of 802.11 MAC and physical layers. WARP Board 1 is connected through Ethernet cable to a Lenovo T60; it functions as the wireless interface of the latter. WARP Board 2 is connected to a Dell Latitude D610 and functions as the access point. This configuration is limited in that the access point is not really a gateway and does not have the capability to forward data to the Internet. Therefore the mobile station is not connected with the Internet, limiting the benchmark applications that can be used to evaluate µPM.

Power-Saving Mode: We implement a single power-saving mode for WARP by completely shutting down the RF transceiver on the radio board. Our measurement shows that the RF transceiver takes far less than 100µs to turn on and the minimal profitable idle interval to turn it off is about 100µs. The whole radio daughter board consumes 3.55 and 2.65Watt when the transceiver is idle and shutdown, respectively. It is important to note that the WARP board does not represent commercial 802.11 network cards in its power profile due to its focus on openness and programmability. For example, the FPGA dominates the power consumption of the whole system (8 Watt out of 12 Watt) without any power management support. In commercial implementation of 802.11 wireless interfaces, the transceiver usually dominates in power consumption.

It is important to note that the RF transceiver is the bottleneck to power management, because the frequency synthesizer in it will dominate the wakeup latency and energy [1] if it is shutdown. Our implementation, however, shows that the entire transceiver can be shutdown during active data transfers to save energy using µPM. Therefore, µPM can readily benefit from power-saving modes with more components managed, including the baseband processor on which it can be implemented.

Measurement: To measure the dynamic power consumption of the radio board, we power the radio board with a separate power supply and connect a 0.1ohm sense resistor in series with it. We calculate the power consumption of the radio board based on the



(a) Consistent Power Reduction

(b) Data Loss Rate (No Consistent Difference)

Figure 13: Measurement for WARP implementation

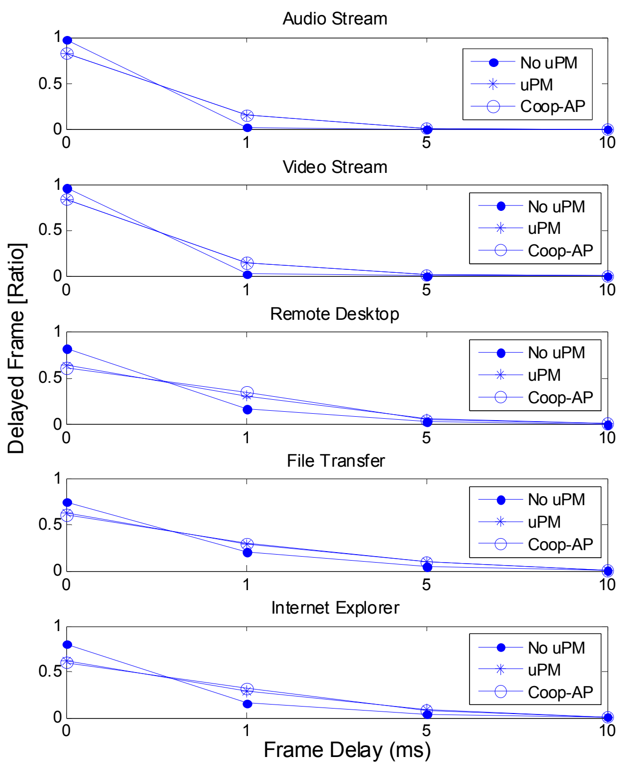


Figure 14: Delayed frame ratios

voltages at the two ends of the sense resistor, which are measured by a 2-channel voltage sampling device at 10KHz. To get the power consumption of the RF transceiver only, we first measure the base power consumption of the radio board when the RF transceiver is turned off and then subtract it from the total power consumption of the radio board. We repeat each measurement multiple times and report the average. We employ Wireshark [24] on host laptops to monitor network traffic and have developed software to record frame delay and loss.

µPM Implementation: We implement µPM as summarized in Section 5.5 in the PowerPC core and employ library functions provided by WARPMAC framework to shut down the RF transceiver. The programmability of WARP also allows us to evaluate the cooperative access point solution as discussed in Section 5.4.

Benchmark Applications: We employ five applications that produce network traffic of different natures. The first two applications are based on VideoLAN [28], a multi-platform media player with capability of sending and receiving media stream over network. We set up a 128Kbps audio stream and 367Kbps video stream from the access point to the wireless interface. The third application is Windows Remote Desktop Connection, which generates burst traffic between the access point and the wireless interface with µPM. The fourth application is transferring a large file over FTP connection. It creates a consecutive traffic flow at high data rate. We set up an FTP server on the access point using CrossFTP and let the mobile station with µPM connect to it and download a big file. The peak downloading data rate is between 560Kbps and 800Kbps, limited by 2Mbps Physical layer data rate on WARP. The last application is Internet Explorer, which generates burst traffic with contents in various media and sizes. We establish a HTTP server on the access point through KF Web Server and publish a website built from Microsoft Office Publisher Template containing multiple pages and plenty of images.

6.2 Experiment Results

Figure 13 presents the measured average power consumption in the 802.11 transceiver and data loss rates for the five benchmarks with and without µPM, and Coop-AP. It shows that µPM and Coop-AP achieve over 30% power reduction without statistically significant difference in data loss rate. Our measurements show that all frame delays are below 5ms and Figure 14 presents the ratios of delayed frames for the five benchmarks with and without µPM, and Coop-AP. It shows that µPM and Coop-AP do not increase frame delays over 10ms for all five benchmarks. For Audio and Video Streaming, the ratio of delays over 1ms are extremely low and µPM only increases it slightly, likely because of their periodic or more predictable traffic. Similarly, File Transfer experiences minor increase in the ratio of frame delays over 1ms with µPM since its traffic is also very predictable. However, File Transfer sees longer frame delays than Audio and Video Streaming because the frame size in File Transfer is larger and the probability of frame loss is greater. In contrast, Remote Desktop and Internet Explorer experience a higher ratio of frame delays over 1ms, likely due to its burst traffic patterns. Again, µPM and Coop-AP only slightly increase the ratio of frame delays between 1ms and 10ms. Also an informal user

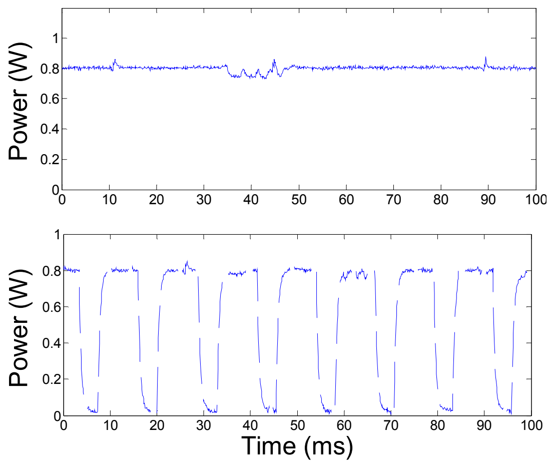


Figure 15: Power traces for VideoLAN-Audio with (bottom) and without µPM (top) (Note they are not synchronized)

evaluation shows no human-perceptible difference either. To provide a concrete idea on how µPM works, Figure 15 shows the power traces of the 802.11 transceiver with and without µPM under VideoLAN audio stream. It shows that µPM is able to shut down the transceiver for a significant portion of the idle intervals during active data transfers.

7. DISCUSSION

The effectiveness of µPM is essentially limited by its compliance to 802.11. First, the retransmission mechanism leveraged to control data loss sets limit to the maximal time and minimal time to spend in an unreachable and the listen sub-state, respectively. This prevents µPM from saving more percentage of energy for lower data rates, as showed in Figure 9. Second, to avoid negative impact on peers, µPM assesses the traffic while in the listen sub-state and applies unreachable sub-states with a decreasing probability when the traffic load increases. This, however, limits µPM's effectiveness in a lightly loaded wireless interface with a busy access point, as demonstrated in Figure 9. Third, and most importantly, missing transmissions by µPM increases data loss under realistic channels as showed in Figure 8(a).

In this work, we show that a minimal modification to the access point implementation can successfully address the data loss problem by adding several extra retransmissions, henceforth cooperative AP, as showed in Figure 8(b). More changes to 802.11, however, is necessary to further achieve the potential of µPM. Ideally, if the wireless interface with µPM can notify the access point how long it will be unreachable, the access point can hold the next frame(s) properly thus achieve the same data loss without extra retransmissions. Such holding will prevent the access point from being engaged in a frame exchange with an unreachable wireless interface thus make it available for serving the other nodes. Note that the energy consumption for transmitting one extra byte is negligible as compared to the energy saving. For example, it is about the same as that by being idle for one microsecond for the PRISM 802.11 transceiver.

We also realize that the optimal number of retransmissions to miss may depend on the channel condition and traffic pattern. In the current implementation, a static allowable number of (four) misses of retransmissions is employed. An investigation on combining channel condition estimation, traffic prediction, and energy models may yield a better, dynamic limit for retransmission misses.

While we demonstrate the feasibility and effectiveness of reducing power consumption of active 802.11 interfaces by carefully switching them between power-saving and idle modes, our evaluation is by no means exhaustive, as 802.11 is complicated and can be applied with a variety of settings. For example, some 802.11 networks employ Auto Rate Fallback (ARF) [27], which reduces the data rate of retransmissions for reliability and therefore may reduce the effectiveness of µPM.

8. CONCLUSIONS

Using traces from real network usage, we show that an active 802.11 interface can spend most time and energy in brief idle intervals. Such short intervals are out of the reach of the standard 802.11 PSM, which is effective only for elongated idle periods.

We present micro power management (µPM) as a standard-compliant solution to reduce the power consumption in such short intervals. µPM leverages the low-overhead power-saving modes supported by modern 802.11 transceivers and the retransmission mechanism of 802.11. It puts an 802.11 interface into unreachable power-saving modes between two frames, guaranteeing constraints in data loss and delays. The key technical components of µPM include frame interval prediction and adaptation to guarantee delay constraints and control impact on the network throughput.

We present an efficient yet effective implementation of µPM. Simulations based on NS-2 and traces from realistic usage demonstrate the effectiveness of the proposed implementation. With minimal cooperation from the access point, µPM can notify the access point about its decision for power management so that the latter will attempt extra retransmissions to compensate the missed ones, thus guaranteeing no additional data loss. Our evaluations show that the efficient implementation effectively adapts to channel condition, to demanded throughput, and to the load of the access point. µPM is able to save considerable energy under modest load on the interface and the access point while managing a very small impact on throughput.

We report a prototype of µPM based on the open-access WARP platform and document the power consumption, frame delay, and data loss when a µPM-enabled WARP board is used as the 802.11 interface for a laptop computer for realistic benchmark applications. Our measurement shows more than 30% power reduction in the wireless transceiver without any statistically meaningful increase in data loss or delay.

9. ACKNOWLEDGEMENTS

The authors would like to thank Dr. Ashutosh Sabharwal, Patrick Murphy, Chris Hunter, and Charles Camp for their help with the WARP platform. The work was supported in part by NSF awards CNS/CSR-EHS 0720825, CNS/NeTS-WN 0721894, and support from TI Leadership University Innovation Fund. The WARP platform was supported by NSF award CNS-0551692. The authors would like to thank our reviewers and our Shepherd, Professor Satyanarayanan, whose comments and shepherding have helped improve this paper.

10. REFERENCES

[1] Andren, C., Bozych, T., Rood, B., and Schultz, D. Intersil: PRISM power management modes: Application Note AN9665, February 1997.

[2] Acquaviva, A., Simunic, T., Deolalikar, V., and Roy, S. Remote power control of wireless network interfaces. *J. Embedded Computing*, August 2005.

[3] Bertozzi, D. Benini, L. Ricco, B. Power aware network interface management for streaming multimedia. In *Proc. IEEE Wireless Communications and Networking Conf.*, March 2002.

[4] Biswas, S.; Datta, S. Reducing overhearing energy in 802.11 networks by low-power interface idling. *Int. Conf. Performance, Computing and Communications*, April 2004.

[5] Brakmo, L. S., Wallach, D. A., and Viredaz, M. A. µSleep: a technique for reducing energy consumption in handheld devices. In *Proc. Int. Conf. Mobile Systems, Applications, and Services* (*MobiSys*), June 2004.

[6] Costa-Perez, X. and Camps-Mur, D. AU-APSD: Adaptive IEEE 802.11e unscheduled automatic power save delivery. In *Proc. IEEE Int. Conf. Communications*, June 2006.

[7] Chen, Y., Smavatkul, N., and Emeott, S. Power management for VoIP over IEEE 802.11 WLAN. In *Proc. IEEE Wireless Communications and Networking Conf.*, March 2004.

[8] Chandra S. and Vahdat A. Application-specific network management for energy-aware streaming of popular multimedia formats. In *Proc. USENIX Annual Technical Conf.*, June 2002.

[9] Gast, M. *802.11 Wireless Networks: The Definitive Guide*. O'Reilly Media Inc., 2005.

[10] Gupta, M., Grover, S., and Singh, S. A feasibility study for power management in LAN switches. In *Proc. IEEE Int. Conf. Network Protocols* (*ICNP*), October 2004

[11] Jensen, J. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica* 30: 175-193, 1906.

[12] Krashinsky, R. and Balakrishnan, H. Minimizing energy for wireless web access with bounded slowdown. *J. Wireless Networks* 11, 1-2, January 2005.

[13] Maxim Integrated Product. Data sheet for MAX2829 Single-/Dual-Band 802.11a/b/g World-Band Transceiver ICs.

[14] Nathawad, L., Weber, D. Abdollahi, S., Chen, P., Enam, S., Kaczynski, B., Kheirkhahi, A., Lee, M., Limotyrakis, S., Onodera, K., Vleugels, K., Zargari, M., and Wooley, B. An IEEE 802.11a/b/g SoC for Embedded WLAN Applications. In *Digest of Technical Papers of IEEE Int. Conf. Solid-State Circuits*, February 2006.

[15] Pering, T., Agarwal, Y., Gupta, R., and Want, R. CoolSpots: reducing the power consumption of wireless mobile devices with multiple radio interfaces. In *Proc. Int. Conf. Mobile Systems, Applications and Services* (*MobiSys*), June 2006.

[16] Paxson, V. and Floyd, S. Wide-area traffic: the failure of Poisson modeling. In *IEEE Trans. on Networking*, June 1995.

[17] Qiao, D. and Shin, K.G. Smart power-saving mode for IEEE 802.11 wireless LANs. In *Proc. Ann. Joint Conf. IEEE Computer and Communications Societies* (*INFOCOM*), March 2005.

[18] Rohl, C., Woesner, H., Wolisz, A. A Short Look on Power Saving Mechanisms in the Wireless LAN Standard Draft IEEE 802.11. In *Proc. WINLAB Wrkshp. Third Generation Wireless Systems*, March 1997.

[19] Rahmati, A. and Zhong, L. Context for Wireless: Context-sensitive energy-efficient wireless data transfer. In *Proc. Int. Conf. Mobile Systems, Applications, and Services* (*MobiSys*), June 2007.

[20] Simunic, T., Benini, L., Glynn, P., and De Micheli, G. Dynamic power management for portable systems. In *Proc. Int. Conf. Mobile Computing and Networking* (*MobiCom*), August 2000.

[21] Shih, E., Bahl, P., and Sinclair, M. J. Wake on wireless: an event driven energy saving strategy for battery operated devices. In *Proc. Int. Conf. Mobile Computing and Networking* (*MobiCom*), September 2002.

[22] Terrovitis, M., Mack, M., Singh, K., Zargari, M. A 3.2 to 4 GHz, 0.25 µm CMOS frequency synthesizer for IEEE 802.11a/b/g WLAN. In *Digest of Technical Papers of IEEE Int. Conf. Solid-State Circuits*, February 2004.

[23] Wireless Open-Access Research Platform, http://warp.rice.edu, 2007.

[24] Wireshark: http://www.wireshark.org/.

[25] Hernandez-Campos, F. and Papadopouli, M. Assessing the real impact of 802.11 WLANs: a large-scale comparison of wired and wireless traffic. In *Proc. IEEE Wrkshp. Local and Metropolitan Area Networks*, September 2005.

[26] Zhong, L. and Jha, N. K. Dynamic power optimization targeting user delays in interactive systems. In *IEEE Trans. on Mobile Computing*, November 2006.

[27] Van der Vegt AJ. Auto rate fallback algorithm for the IEEE 802.11a Standard. *Technical report*, Utrecht University, 2002.

[28] Video LAN, http://www.videolan.org/.

APPENDIX: Wireless Frame Trace Collection

We collected application-specific traces from a Windows XP-based Dell Latitude 610 laptop with 802.11g interface using Wireshark [24] from a home 802.11g network wired through AT&T DSL Elite service (up to 6Mbps according to AT &T). The traces last between 5 to 20 minutes. We also collected the complete 802.11 network traffic from four Windows laptops for one week. All four users are from Rice University and they used wireless network extensively at both school and home.

We used Wireshark because it enabled us to collect traces with minimal intrusion to the users. However, there is inherent inaccuracy in its logging due to OS timing granularity and interrupt handling. We consider such inaccuracy acceptable for evaluating the idle interval prediction by µPM because the likely inaccuracies are in the order of ten microseconds and are random in nature. In addition, the logger does not record control frames. Again, we consider it acceptable for the following reason. First of all, µPM does not power manage the wireless interface until the exchange of control frames finishes. Then the absence of control frames only introduces inaccuracy to idle interval estimation. Fortunately, the intervals between control frames and their transmission time are almost constant. Therefore the inaccuracy in idle interval estimation can be reduced by estimating the time overhead of control frames.