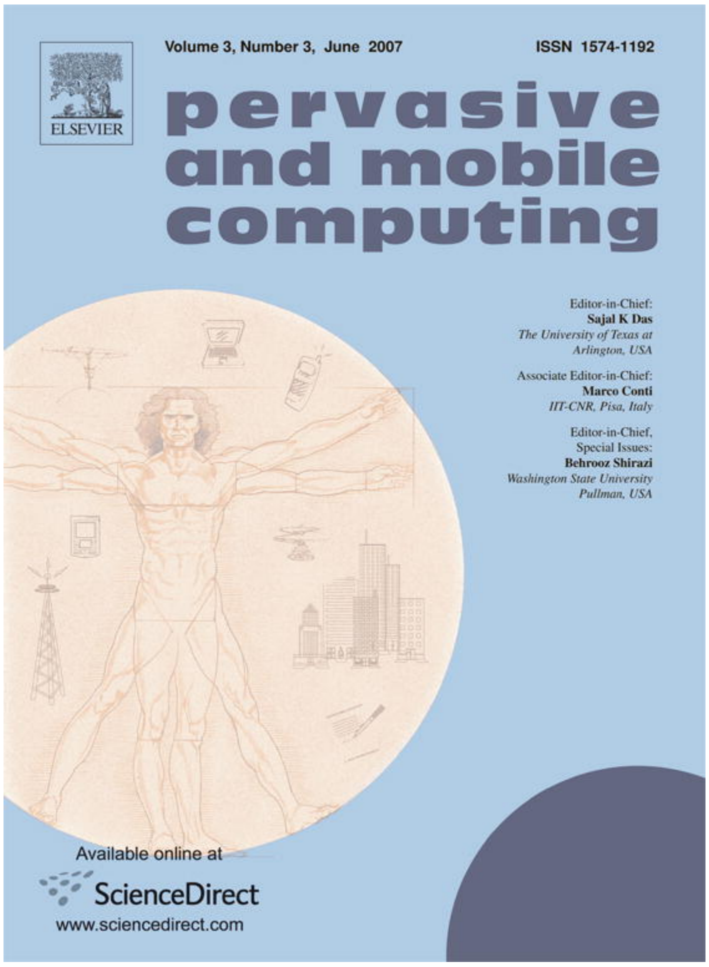

Volume 3, Number 3, June 2007
ISSN 1574-1192
ELSEVIER
pervasive and mobile computing
Editor-in-Chief: Sajal K Das The University of Texas at Arlington, USA
Associate Editor-in-Chief: Marco Conti IIT-CNR, Pisa, Italy
Editor-in-Chief, Special Issues: Behrooz Shirazi Washington State University Pullman, USA
Available online at ScienceDirect www.sciencedirect.com

Available online at www.sciencedirect.com
ScienceDirect
Pervasive and Mobile Computing 3 (2007) 255–275
pervasive and mobile computing
www.elsevier.com/locate/pmc

Adaptive application-driven WLAN power management

Yu Jiao[a], Ali R. Hurson[a,*], Behrooz Shirazi[b]

[a] *Department of Computer Science and Engineering, The Pennsylvania State University, United States*
[b] *School of Electrical Engineering and Computer Science, Washington State University, United States*

Received 29 April 2005; received in revised form 25 October 2006; accepted 22 December 2006
Available online 25 January 2007

Abstract

In this paper we present an adaptive application-driven power management (AADPM) strategy with online idle period length distribution learning capability for the IEEE 802.11b WLAN. We discuss its design and evaluate the performance in comparison with other power management strategies using the network simulator NS2. We simulated both the single user and multiple user scenarios. Experimental results have shown that, compared with other power management methods examined in this paper, AADPM achieved the highest energy saving in all cases and it demonstrated strong adaptability to network congestion.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Energy-aware systems; Mobile communication systems; Algorithm/protocol design and analysis

1. Introduction and motivation

Battery-powered systems are often confronted with the problem of delivering high performance with limited computational and energy resources. Wireless networks provide mobile users the opportunities of remote information access and sharing. However, without

* Corresponding author.
E-mail addresses: yjiao@cse.psu.edu (Y. Jiao), hurson@cse.psu.edu (A.R. Hurson), shirazi@wsu.edu (B. Shirazi).

1574-1192/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.pmcj.2006.12.003

an effective power management (PM) strategy, the wireless network interface card (NIC) can quickly drain the battery of the mobile device. Kravets and Krishnan have shown in their paper that the NIC may consume up to 10% of the total energy of a high-end portable computer and nearly 50% of the overall energy of a low-end handheld device [8].

Our goal is to minimize the energy consumption of the wireless NIC while maintaining high throughput for wireless request/response type applications. The target environment is the popular IEEE 802.11 infrastructure wireless LAN (as opposed to the IEEE 802.11 ad hoc network) [4], in which all stations communicate with an access point (AP). When the 802.11b power save mode (PSM) is enabled, the AP broadcasts a *Beacon* every *Beacon Period* (typically 100 ms). During the BeaconPeriod, the AP buffers all data destined for the stations. Each Beacon contains a traffic indication map (TIM) showing whether a station has data buffered at the AP. IEEE 802.11b compliant wireless NICs support two power states: *awake* and *doze*. The NIC is fully powered while awake. In the doze state, the NIC is not able to transmit or receive packets and consumes very low power. When the PSM is enabled, a mobile station can switch its NIC to doze state when it has no data to send or receive in order to conserve energy. The station wakes up periodically and listens to the Beacon. If the TIM indicates that there are data buffered at the AP, the mobile station must poll the AP and receive all outstanding data. Typically, a station listens to every Beacon, but it can be configured to skip Beacons. PSM is often referred to as a static power management scheme because of the fixed listening interval.

Research has shown that while PSM does quite well in saving energy in some cases, it is not flexible enough to balance energy consumption and throughput [1,7]. The round trip time (RTT) is often used as the performance metric for request/response type applications. PSM is too coarse-grained for applications that have RTTs much shorter than one BeaconPeriod. If the BeaconPeriod is 100 ms, depending on when a request is issued relative to the Beacon, PSM may introduce a maximum of 100 ms and an average of 50 ms delay to the request (see Fig. 1(a)). This delay is often not acceptable to latency sensitive applications. On the other hand, PSM is too fine-grained for applications that have long server response time. As shown in Fig. 1(b), for a request that has an RTT of about 800 ms, the station wakes up and listens to the Beacon nine times before it receives the response. We must note that waking up and listening to Beacons are not free in terms of energy consumption. If the energy saving during the doze state cannot amortize the energy consumed during the wakeup and listening to Beacon periods, PSM may even cause more energy consumption than when there is no power management (NO_PM) [1].

In this paper, we propose and evaluate an application-driven dynamic power management (DPM) approach that has online learning capability. It requires no changes to the IEEE 802.11b protocol. In addition, its effectiveness is independent from the input because it does not assume a particular network traffic pattern. In order to evaluate our scheme, we extended the network simulator NS2 [11] to support 802.11b PSM and compared the performance of the proposed scheme with several other PM strategies reported in the literature. We present the results of topologies that contain a single mobile station and multiple mobile stations. To the best of our knowledge, our work is the first simulation study on the adaptability of DPM methods in the presence of multiple mobile stations.

The rest of this paper is organized as follows: Section 2 introduces the related work; Section 3 discusses the design and implementation of our adaptive application-driven DPM

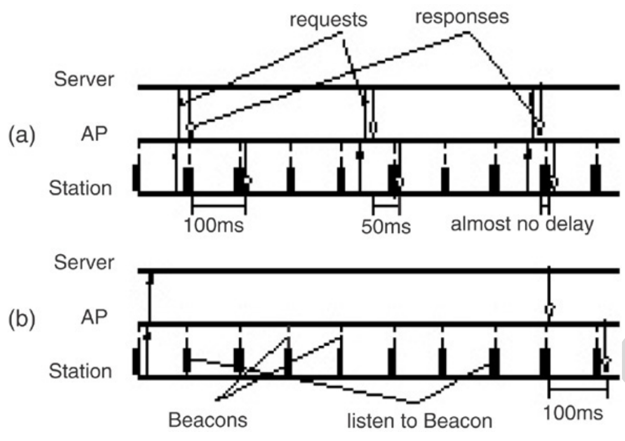


Fig. 1. Static PSM.

(ADDPM) strategy; Section 4 presents the experimental results and analysis; and finally, Section 5 concludes our work.

2. Related work

Observations of the inflexibility of the static PSM have led to the development of many dynamic power management (DPM) strategies [1,2,5,7,8].

Cisco Aironet 802.11a/b/g wireless LAN adapters provide a DPM option, which switches the NIC between the power save mode (PSM) and the continuous awake mode (CAM) depending on network traffic [3]. It switches to CAM when retrieving a large number of packets and switches back to PSM after the retrieval. This DPM approach switches the NIC to CAM when more than one packet is waiting at the AP, and it switches the NIC back to PSM after approximately 800 ms of inactivity [1]. We refer to this approach as the fixed-timeout DPM (FTDPM) for the rest of the paper.

Krashinsky and Balakrishnan proposed the Bounded-Slowdown (BSD) protocol [7], which operates completely at the link layer with no higher-layer knowledge. Under the assumption that NIC power management mode transitions between CAM and PSM are always successful and introduce negligible overhead, the authors mathematically proved that the BSD protocol guarantees that the observed RTT is less than $(1 + p)$ times longer than the RTT measured in the absence of PSM, where p is the expected factor of RTT delay ($p > 0$). Experimental results show that in many cases, the BSD protocol can significantly reduce the energy consumption of web page retrieval while guaranteeing the performance bound. Note that the assumption regarding the NIC power management mode transition made by the authors may not hold in reality [1]. As we will show in the evaluation section, violations of this assumption may lead to RTTs significantly out of bounds.

Compared to the low-level PM protocols discussed above, application-driven PM approaches have the advantage of knowing the application's intention of network usage. This knowledge provides more opportunities for energy saving without sacrificing throughput. However, unlike low-level PM, there is no one-size-fits-all solution to application-level PM, because applications may exhibit drastically different network traffic patterns. Many application-level PM protocols have been developed to embrace the energy/throughput tradeoff [1,8].

The protocol proposed by Kravets and Krishnan is a general application-level DPM [8], in which the mobile station acts as the master and the AP is the slave. The slave only sends out data when the master commands it to do so. Both the master and slave can queue up data during non-transmit phases, in which the NIC is suspended to save energy. This protocol uses a fixed *timeout period* to determine whether or not to suspend the NIC, and it applies the *sleep duration* parameter to control the length by which the NIC should be suspended. Performance evaluation on Web and Email applications shows that this approach conserves 83% of the energy consumed by communication at the cost of a delay in RTT that ranges from 0.4 to 3.1 s. This protocol has two drawbacks. First, it requires changes to the AP and stations' firmware, because currently, 802.11b compliant APs cannot act as the slave of a station. Second, selfish mobile stations may ask the AP to queue large amount of packets for itself and therefore, cause overflow and packet losses at AP.

The self-tuning power management (STPM) scheme has been developed to improve the energy efficiency and reduce the RTT delay of latency sensitive applications [1]. It uses the history of a group of requests that are closely related in time (150 ms) to determine whether or not to change the NIC power mode. Results show that STPM can reduce the energy usage of a distributed file system application by 21% compared to PSM, while reducing the RTT delay by 80%. However, STPM is not suitable for request/response type applications that involve long user think time (e.g. web browsing) or applications that have long server response time (e.g. distributed information retrieval), because they cannot generate enough traffic for STPM to switch to CAM. As a result, STPM will behave the same as PSM for those applications.

When a NIC is in the doze mode, it is shut off from the rest of the network. The effectiveness of the DPM strategies discussed in this section depends on how well they predict the ongoing activities in the network during their connectivity blackout. Wake-on-wireless [10] works as a nice complement to existing DPM methods by using a low-power network to signal a mobile station when packets are waiting at the AP. However, this technology is not yet widely available.

We proposed an adaptive application-driven power management (AADPM) strategy with online idle period length distribution learning capability. The work that is the most similar to ours is the STPM protocol [1]. However, our primary targets are distinctively different. STPM focus on latency sensitive, interactive applications such as distributed file systems where system response time is short in comparison to the BeaconPeriod, while AADPM addresses request/response type applications that involve long server response time and long user think time. AADPM differs from other DPM methods in several ways:

- Unlike some protocols [7,8], it requires no changes to the existing IEEE 802.11b protocol.
- It does not assume a particular request arrival pattern, nor does it make assumptions about the server response time. It learns the idle period length distribution online.
- Unlike the BSD protocol [7], it explicitly considers the mode transition cost when making transition decisions.

- None of the previously mentioned PM strategies explicitly consider the possibility of having multiple NIC low-power states. When a NIC supports multiple low-power states, our scheme maximizes energy saving by intelligently choosing the most suitable state.

3. Adaptive application-driven power management (AADPM)

In this section, we characterize NICs, discuss energy-aware application design, and present the AADPM policy.

3.1. NIC characterization

Let's first differentiate two concepts: the operational states from the power management modes of a NIC. The operational states refer to the fact that a NIC can be operated in either a full power consumption status (awake) or a low power consumption status (doze). The power management modes govern the switch between these two operational states. If the low power operational state (doze) is disabled, the NIC is said to be in a Continuous Awake power management Mode (CAM). If the low power operational state is enabled, on the other hand, the NIC is said to be in a Power Save Mode (PSM) because the doze state can be utilized to conserve energy when appropriate. If the AP knows that a mobile station is in CAM, it will forward all packets destined for the mobile station without delay. If the AP knows that a mobile station is in PSM, it will store all packets destined for the mobile station until the station polls for them.

A mobile station can change its operational state at anytime and such operational state transition takes a fixed amount of time; however, it cannot switch between the power management modes, CAM and PSM, at will. It must notify the AP of the power management mode change intention because the AP will treat packets destined for the mobile station differently according to the station's current power management mode. According to the 802.11b specification, a power management mode change can be carried out only after a successful frame exchange between the mobile station and the AP. A frame exchange typically includes a RTS (request-to-send) packet, a CTS (clear-to-send) packet, a data packet, and an ACK packet. Thus, it is not guaranteed that a power management mode change between CAM and PSM will always be successful, and the overhead introduced by such a transition depends on the network condition. It is reported that a power management mode transition may take as long as 600 ms [1].

As reported in [7], the Enterasys Networks RoamAbout NIC consumes 750 mW in the awake state (transmitting data, receiving data, or idle), and 50 mW while it is in the doze state. The operational state transition from doze to ready to receive (awake) takes 2 ms, and the energy consumption is 1.5 mJ (750 mW being consumed for 2 ms). The time overhead caused by the power management mode transition, $t_{\text{pm\_mode\_change\_overhead}}$ cannot be predetermined and it is measured during the simulation. We used these NIC parameters in all our experiments.

Note that operational state transition from awake to doze usually consumes negligible energy, but the reverse is not true. An operational state transition from awake to doze will save energy only if the energy saving in the doze state can amortize the energy required for the state transition from doze to awake.

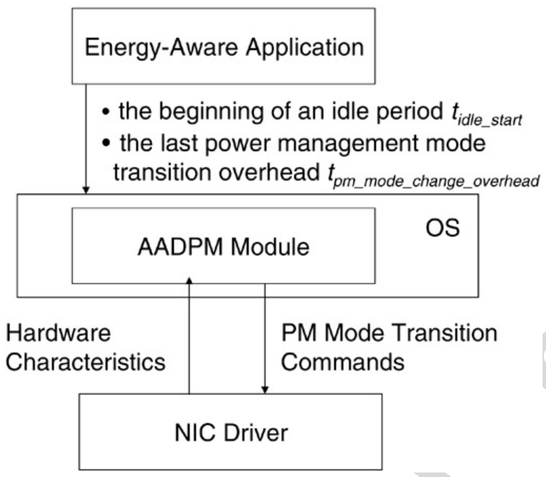

Energy-Aware Application
• the beginning of an idle period t_{idle_start}
• the last power management mode transition overhead $t_{pm\_mode\_change\_overhead}$
AADPM Module
OS
Hardware Characteristics
PM Mode Transition Commands
NIC Driver

Fig. 2. The AADPM architecture.

Let P_{awake} and P_{doze} denote the power usage of a NIC in the awake and doze state, respectively. $E_{\text{operational\_state\_transition}}$ represents the energy consumption during the operational state transition from doze to awake. We define a break-even time t_{BE}, which is the minimum amount of time that the NIC needs to stay in the doze state in order to achieve energy saving. The t_{BE} is calculated by Eq. (1). The t_{BE} of the above Enterasys Networks Roam About NIC is approximately 2 ms.

$$t_{\text{BE}} = \frac{E_{\text{operational\_state\_transition}}}{P_{\text{awake}} - P_{\text{doze}}}. \tag{1}$$

3.2. Modifications to existing applications

As an application-level power management scheme, AADPM does not require changes to the firmware. Instead, a software module, the AADMP module, is introduced between the application and the NIC driver. It receives information from the application and controls the NIC through the driver software. This is a design similar to the one proposed by Anand et al., and the authors have shown how to implement it as a Linux kernel module [1]. Fig. 2 illustrates the AADPM architecture.

In order to communicate with the AADPM module, we need to make some fairly simple modifications to existing applications: (i) the server sends an application-level ACK to the client when a request is accepted, (ii) the client program notifies the AADPM Module when an application-level ACK or a response is received, and (iii) the client program gathers statistics about the idle period history and the last mode transition cost. We refer to applications that satisfy these three requirements as energy-aware (EA) applications.

The first requirement ensures that the application is transport-layer independent — its function does not depend on a reliable transport-layer protocol such as TCP. The second requirement provides the AADPM module the precise knowledge of the beginning of an idle period. The third requirement allows the AADPM to predict the next idle period length. More specifically, we use two separate histograms to record the lengths of the server response time and user think time. The idle period length probability distribution can be calculated using the histograms, which in turn can be used by the AADPM to generate a probability-based idle period length prediction.

The algorithm that creates the histograms can be formally stated as follows. The set of all possible idle period lengths $(0, \infty)$ is partitioned into n intervals, where n is the number of bins in the histogram. Let l_i denote the starting point of interval i $(0 \leq i < n)$. Bin i represents the idle period length range $\in [l_i, l_{i+1})$. If an idle period has a length greater than ln, it belongs to bin $n - l$. Each bin has a counter C_i that indicates the number of idle period lengths that falls in the range of bin i.

A history window size ω is used to define the total number of history records — only the most recent ω idle periods are recorded ($\sum_{i=0}^{n-1} Ci = \omega$). The probability that the length of the next idle period will fall in the range of bin i is $p(i) = \frac{Ci}{\omega}$. The cumulative distribution function (CDF) is $P(i) = \frac{\sum_{j=0}^{i} Cj}{\omega}$.

3.3. AADPM

3.3.1. Design principles

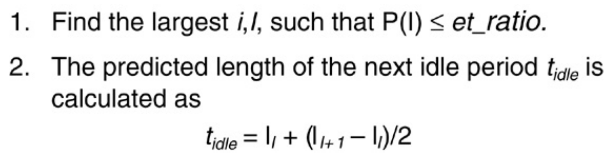
The design principles of the AADPM strategy are as follows:

- Do not assume a particular idle period distribution. Instead, we use online distribution learning. This avoids the problem that the PM strategy works well if the reality matches the assumption, but will perform poorly otherwise.
- Allow the user to provide an energy/throughput tradeoff indicator, the *et_ratio*, as the guideline for predicting the upcoming idle period length. The et_ratio ranges from 0 to 1.0 with 0 giving maximum throughput (minimum RTT delay, in other words) and 1.0 maximum energy conservation. If no et_ratio is specified, the default value used is 0.5, where equal priority is given to energy saving and performance.
- Explicitly consider the power management mode transition cost, $t_{\text{pm\_mode\_change\_overhead}}$, when making mode transition decisions. The AADPM module makes a decision for the NIC to transit between CAM and PSM only when it believes that after subtracting the power management mode transition cost from the predicted idle period length, the idle period is still longer than the break-even time.

3.3.2. Predict the next idle period length

The predicted upcoming idle period length has a direct impact on the energy/throughput tradeoff: if the prediction is much shorter than the upcoming idle period, the energy saving is not maximized; if it is much longer than the upcoming idle period, the RTT delay increases. Given the CDF of the idle period length, Fig. 3 presents the prediction algorithm.

Note that et_ratio is not a precise bound for RTT delay. It is an indictor of, out of energy saving and throughput, which factor is more important to the user. AADPM uses the et_ratio as a guide when making a prediction. For instance, by setting the et_ratio to 0.8, the user implies that 80% priority is given to energy saving and 20% priority is given to throughput. The actual amount of savings depends on the distribution of the RTT.



1. Find the largest *i*,*l*, such that P(l) ≤ *et_ratio.*
2. The predicted length of the next idle period t_{idle} is calculated as

$$t_{idle} = l_i + (l_{i+1} - l_i)/2$$

Fig. 3. The idle period length prediction algorithm.

3.3.3. The AADPM policy

An EA application discloses two hints to the AADPM module: the beginning of the upcoming idle period $t_{\text{idle\_start}}$ and the last power management mode transition cost $t_{\text{pm\_mode\_change\_overhead}t}$. The AADPM obtains the break-even time t_{BE} and wakeup delay t_{delay} parameters from the NIC. t_{idle} is calculated as shown in Fig. 3.

Assume that the default power management mode of a NIC is CAM. The NIC is switched to the PSM mode, only if $(t_{\text{idle}} - t_{\text{pm\_mode\_change\_overhead}t}) > t_{\text{BE}}$. If the upcoming idle period is a user think-time idle period (the idle period after a response is received) and the mode transition condition holds, the NIC goes to sleep until the arrival of the next request. If the upcoming idle period is a wait-for-server idle period (the idle period after a request is sent) and the mode transition condition holds, the NIC sleep duration is governed by t_{sleep} in Eq. (3) below,

$$t_{\text{beacon}} = \left(\lfloor (t_{\text{current}} + t_{\text{idle}})/t_{\text{bp}} \rfloor + 1\right) \times t_{\text{bp}} \tag{2}$$

$$t_{\text{sleep}} = t_{\text{beacon}} - t_{\text{current}} - t_{\text{delay}} \tag{3}$$

where t_{current}, t_{bp}, and t_{beacon} are the current time, the beacon period, and the next Beacon time, respectively. Once the t_{sleep} time expires, the mobile station wakes up periodically and listens to every Beacon like PSM until pending data are detected. It then switches the NIC to the CAM mode and receives all subsequent data without delay (Fig. 4).

4. Evaluation

We compared the performance and energy impact of NO_PM, AADPM, PSM, FTDPM, and BSD through extensive simulation using the network simulator NS2 [11].

4.1. Simulation methodology

NS2 version 2.26 does not support detailed WLAN simulation, nor does it support the 802.11b power save mode. Based on the 802.11b specification, we extended NS2 to provide functions necessary for PM strategy evaluation. We also implemented a library of PM methods that have been reported in the literature. We believe that our NS2 extension and the PM library can significantly shorten the design and test cycles of future PM strategies. The EA application is implemented as an NS2 application agent. NIC parameters discussed in Section 3.1 are summarized in Table 1.

Note that the energy consumed by the NIC is only part of the overall mobile device energy consumption. While PM strategies save energy for the NIC, they also inevitably increase the total execution time of requests. The prolonged execution time consequently

```
1.  AADPM ($t_{idle\_start}$, $t_{idle}$, $t_{mode\_change\_overheadt}$) {
2.    IF (($t_{idle}$- $t_{mode\_change\_overheadt}$) > $t_{BE}$ ) {
3.      IF (user-think-time idle period) {
4.        Switch the NIC to doze mode and let it sleep till the arrival of the next
          request arrives;
5.        Switch the NIC back to awake mode and send the request;
6.      } ELSE IF (wait-for-server idle period) {
7.        Calculate $t_{sleep}$ using equation (3);
8.        Switch the NIC to doze mode and let it sleep for $t_{sleep}$ with out listening
          to Beacons;
9.        Periodically wakeup the NIC to listen to every Beacon till there are
          data pending at the AP;
10.       Switch the NIC back to awake mode and retrieve all pending data;
11.     }
12.   } ELSE
13.     Let the NIC stay in awake mode;
14. }
```

Fig. 4. The AADPM policy.

Table 1
NIC parameters

NIC state	Power (W)	Wakeup delay (ms)
Active	0.75	0
Doze	0.05	2

increases the base power consumption of the mobile station. Thus, it is important to study the impact of a PM strategy on the overall mobile device energy consumption as well as its effect on the NIC energy consumption. It was measured in [1] that the base power of an HP iPAQ 3870 with 64 MB of DRAM and 32 MB of flash memory is 1.44 W. We used this parameter in our simulation.

The AADPM histogram has $n = 1024$ and $\omega = 20$. Each bin in the response time and the think time histograms represents an interval of 100 ms.

We used the UC-Berkley-Home-IP HTTP traces [9,12] to test the different DPM strategies. The CDFs of request size, server response time, and response size used in the

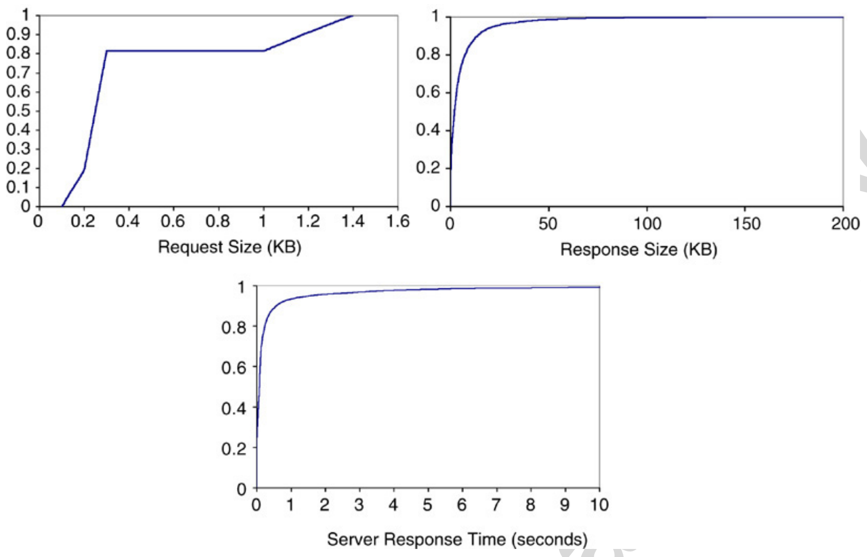


Fig. 5. CDFs of request size, response size and server response time.

experiments are shown in Fig. 5. We used these parameters in all our experiments, unless noted otherwise.

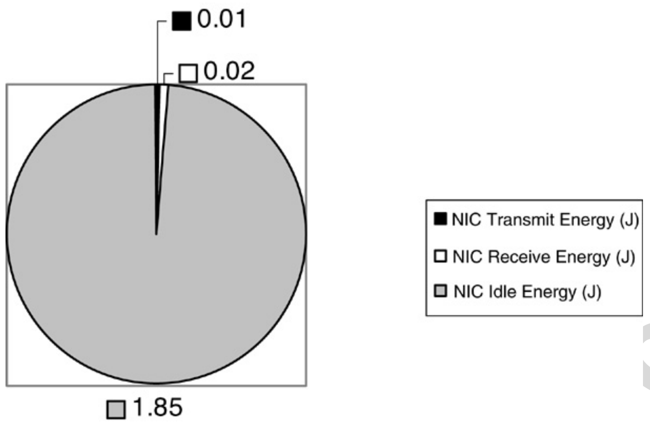
In the performance analysis, we are interested in a group of requests that are closely related to each other in terms of time. However, nearly 80% of the think time intervals obtained from the UC-Berkley traces are longer than 10 s. As a result, the NIC energy consumption during the think time will completely dominate the overall energy consumption, and the performance of static PM methods such as PSM will suffer substantially. In order to provide a fair comparison of different PM strategies, we did not use the trace think time in our experiments. Instead, we used think time intervals that are randomly generated within the range of 1 and 3 s.

The experiment setup includes one or more mobile stations, an AP, and a server. Mobile stations and the AP communicate using the 802.11b protocol. The AP and the server are connected through a duplex link that has a bandwidth of 10 Mbps and a latency of 2 ms.

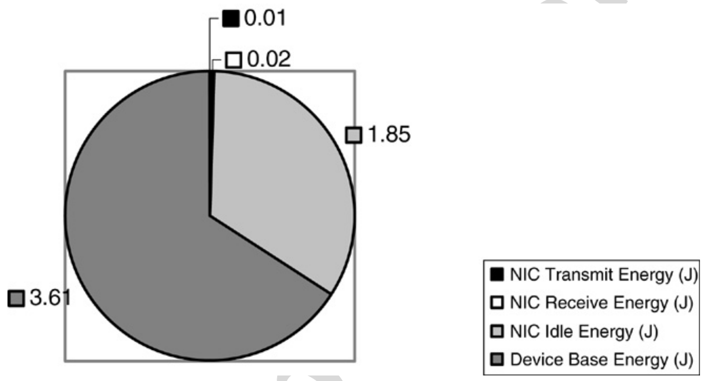
We designed two sets of experiments. In set I, the network topology contains a single mobile station, and the station executes 10,000 requests from the trace. In set II, there are 10 concurrent mobile stations. Each station independently submits 1000 requests from the trace at a randomly generated pace. The purpose of experiment set II is to investigate how well a PM strategy reacts to network congestion.

4.2. *Experiment set I*

Fig. 6(a) presents a decomposed view of the average NIC energy consumption per request. It shows that when no PM is applied, more than 98% of the overall NIC energy consumption is during the idle period. Thus, the effectiveness of a PM strategy depends on whether it can recognize the idle periods and switch the NIC to doze mode when there is a potential energy gain under the performance constraints. Fig. 6(b) plots the overall mobile device energy consumption, which includes the NIC as well as the base energy



(a) NIC energy.



(b) Device energy.

Fig. 6. NIC and the overall device energy consumption with NO_PM (single mobile station).

consumption during the request execution. It shows that the NIC energy consumption only accounts for 34% of the total device energy usage. Thus, in order to achieve the overall energy saving for a mobile device, the energy gain introduced by the NIC must be greater than the base energy loss during the increased execution time.

Fig. 7 compares the performance of NO_PM, PSM, FTDPM, BSD with three different performance bounds — 1.0, 0.5, and 0.1, and AADPM with three different et_ratio values — 1.0, 0.5, and 0.1. When there is not power management scheme (NO_PM), all responses are forwarded to the client immediately without any delay at the AP and therefore, these RTTs are used as the baseline for comparison. When a PM strategy is deployed, the AP will buffer the responses while the station is in doze mode and hence, introduce RTT delays. Fig. 7a shows the RTT delay $((RTT_{PM\_method} - RTT_{NO\_PM})/RTT_{NO\_PM})$ caused by each PM method. Fig. 7b shows the average NIC energy consumption per request for each method.

Results indicate that there is no absolute winner among the PM strategies that we examined: BSD alternatives achieved the shortest RTT delay, but the AADPM variations

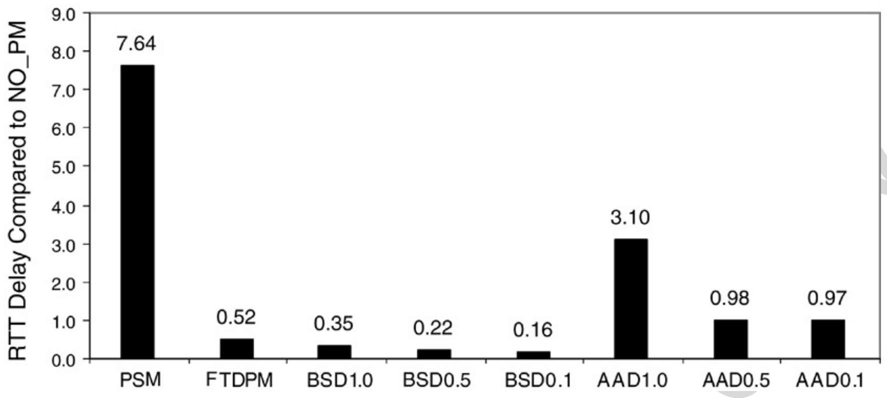


Fig. 7a. RTT delay ratio (single mobile station).

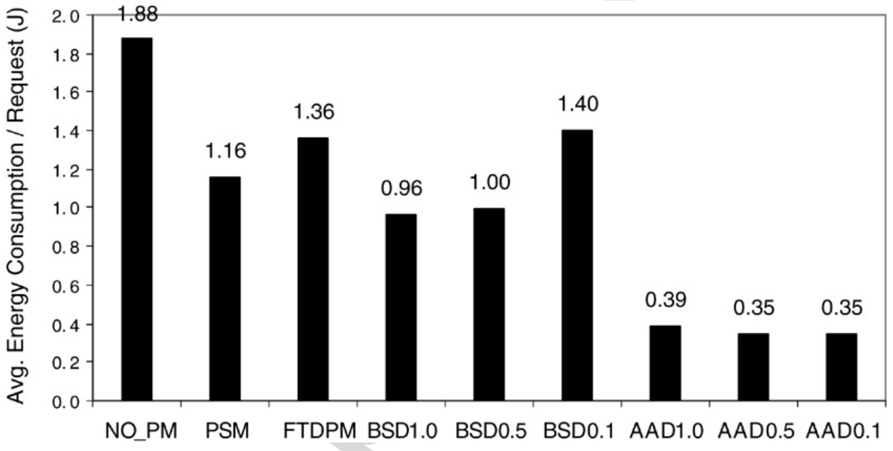
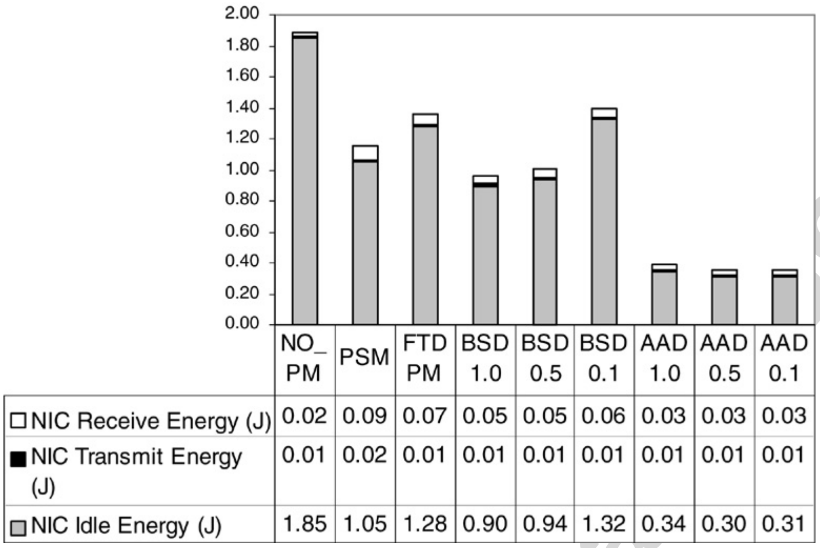


Fig. 7b. Average NIC energy consumption per request (single mobile station).

accomplished the best energy saving. Fig. 8 plots a decomposed view of the NIC energy consumption of each PM method. These results demonstrate that by knowing the application's network usage intention and using the online idle period distribution learning algorithm, AAD0.5 saved 64% more energy than the second-best energy saving PM policy BSD1.0.

Compared to NO_PM, PSM saved 38% of the NIC energy at the cost of a 7.64 times longer RTT delay. Compared to PSM, FTDPM significantly reduces the RTT delay by staying awake for 800 ms and switching back to active mode from doze mode as soon as pending packets are detected. However, the energy consumption of DPM is 17% higher than PSM.

When the proposed AADPM strategy with different et_ratios is applied, the experimental results match our expectation — larger et_ratio leads to more aggressive idle period length prediction, and consequently longer RTT delay. The average delay is 3.1 when the et_ratio is 1.0, and it is 0.97 when the et_ratio is 0.1. However, longer NIC sleep time does not necessarily result in less energy consumption. The average energy consumption is 0.35 J with AAD0.5, while it is 0.39 J with AAD1.0. This is because the



	NO_ PM	PSM	FTD PM	BSD 1.0	BSD 0.5	BSD 0.1	AAD 1.0	AAD 0.5	AAD 0.1
□ NIC Receive Energy (J)	0.02	0.09	0.07	0.05	0.05	0.06	0.03	0.03	0.03
■ NIC Transmit Energy (J)	0.01	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01
▨ NIC Idle Energy (J)	1.85	1.05	1.28	0.90	0.94	1.32	0.34	0.30	0.31

Fig. 8. A decomposed view of the NIC energy consumption (single mobile station).

performance of AADPM depends on the accuracy of the idle period prediction. When the prediction is much longer than the response time, the NIC wastes energy during the overslept period. When the prediction is much shorter than the response time, however, the NIC consumes extra energy by staying awake. Since 61% of the requests from the trace have a response time less than 100 ms, the predicted idle period lengths are more accurate when the et_ratio is 0.5. The fact that most response times are concentrated in a small range also explains why the delay and energy consumption of AAD0.5 and AAD0.1 are nearly the same.

These results suggest that the et_ratio should not be statically chosen. If the response time distribution changes, the ratio should adapt accordingly. One alternative is to use the user specified et_ratio as the starting point. When it is time to make a prediction, multiple et_ratios can be applied and the predictions recorded. The AADPM can choose the et_ratio that generates the most accurate predictions based on its observation over a period of time.

Fig. 7b shows that the energy consumption of AAD0.5 is the lowest among the PM strategies examined. Also the RTT delay of AADPM0.5 is 87% less than PSM, while the energy saving is nearly 70%. However, it does introduce longer delays compared to FTDPM and BSD variations. Compared to FTDPM, AADPM0.5 saved 74% of the energy consumption at the cost of an 88% increase in delay. When compared with BSD0.5, AADPM reduced the energy consumption by 65%, however, it prolonged the RTT delay by 3.5 times. These results imply that AADPM may not be the best PM protocol for applications whose RTTs are dominated by the network delay instead of server response time, for instance, web browsing, where small RTTs may be strongly weighted in importance by the user. The trace used in our experiments showed the near-worst case scenario of the AADPM performance.

We expect AADPM to better demonstrate its advantages in a large body of applications that involves complex remote execution, such as distributed information retrieval that we

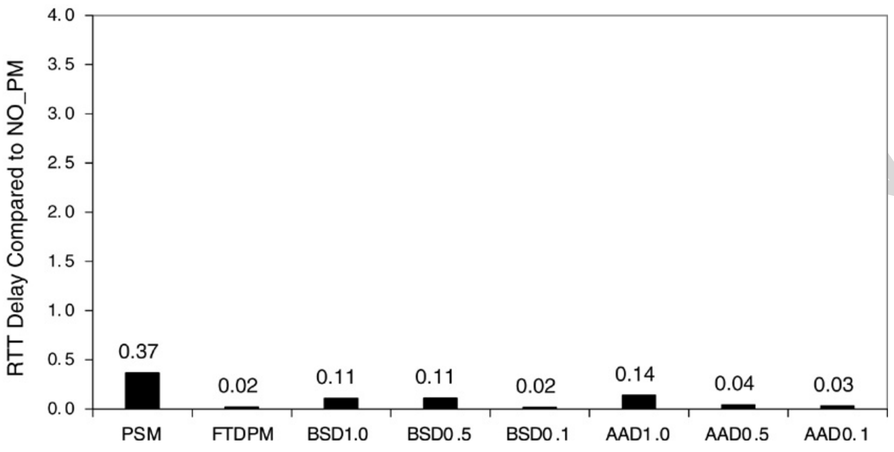


Fig. 9a. RTT delay ratio (single mobile station) with normal response time distribution.

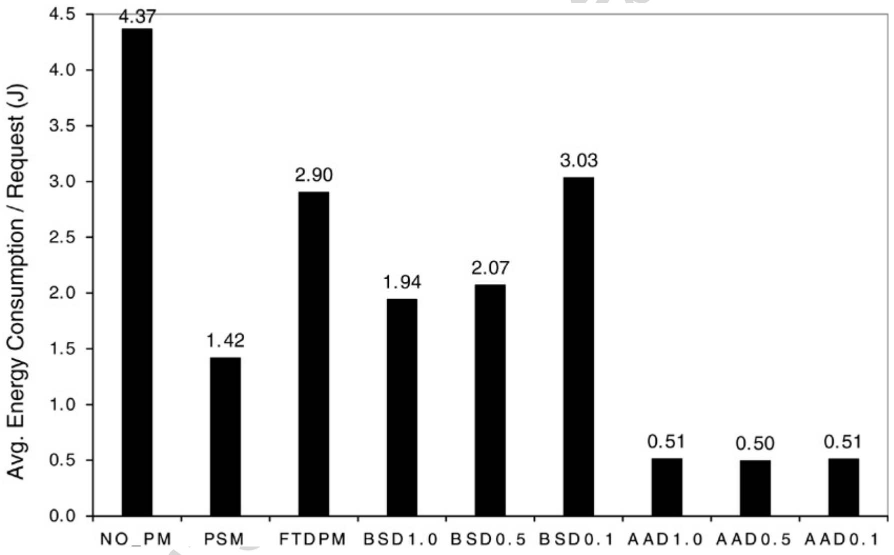


Fig. 9b. Average NIC energy consumption per request (single mobile station) with normal response time distribution.

will discuss next. Moreover, as we will see in the next subsection, AADPM adapts to network congestion better than other PM approaches.

We have prototyped an agent-based wireless distributed information retrieval system called MAMDAS in our previous research [6]. We modelled the server's response time using a normal distribution with $\mu = 2.5$ and $\sigma = 0.2$. Figs. 9a and 9b plot the RTT delay and NIC energy consumption for the different PM approaches. FTDPM and BSD0.1 achieved the best performance at a high cost of energy. AADPM showed performance gains as well as energy savings — compared with PSM and BSD0.5, AAD0.5 achieved an RTT delay reduction of 89% and 64%, respectively; it also improved the energy saving by 65% and 76%, respectively.

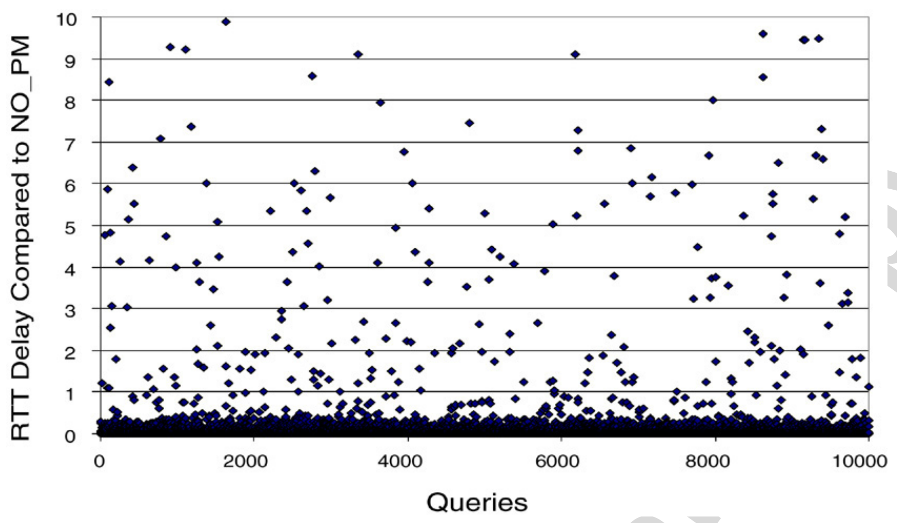


Fig. 10. BSD0.5 RTT delay ratio.

From Fig. 7a we also observe that BSD cannot always guarantee the performance bound — when the BSD performance bound is set to 0.1, the average RTT delay is actually 0.16. Further investigation reveals that when the RTT delay upper bound is 0.5, although the average delay is 0.22, much less than the performance bound, the delay of some requests can be as high as 10 times (Fig. 10).

As we discussed in Section 2, under the assumption that NIC mode transitions are always successful and introduce no overhead, it is mathematically provable that the BSD protocol can guarantee a performance upper bound. However, according to the 802.11b specification [4], mode transition requires one successful frame exchange between the mobile station and the AP. Thus, the success of a transition is not guaranteed, and the overhead is at least one frame exchange. This transition cost can be as high as 600 ms [1]. We believe that it is impossible to provide a hard performance bound in reality. Thus, it is more practical to use an energy/performance tradeoff indicator, such as the et_ratio of ADDPM, in guiding the PM decisions.

Fig. 11 plots the average mobile device energy consumption per request. It shows that the energy reduction of the NIC does not ensure an overall device energy conservation — compared to NO_PM, PSM reduced the NIC energy consumption by 37% (Fig. 7b), however, it increased the overall device energy by 16%. The reason is that the energy consumed by the NIC is only a portion of the total energy used by a device while executing requests. The employment of a PM policy can reduce the NIC energy, but it increases the execution time of a request at the same time. If the base power of a device consumed during the prolonged request execution time cannot be compensated by the NIC energy saving, a PM strategy may result in both performance and energy degradation like PSM did in our experiments.

Fig. 11 also implies that AADPM achieved a good balance between performance and energy. The extra base power consumption caused by the RTT delays is amortized by the NIC energy gain and therefore, AADPM variations also achieved the best overall device

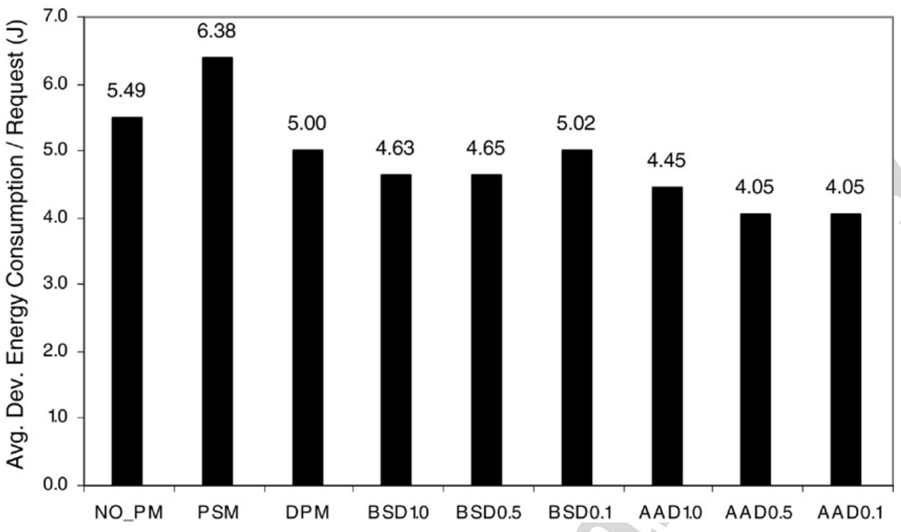


Fig. 11. Average device energy consumption per request (single mobile station).

energy savings. Compared to the approach that achieved the second best overall energy saving — BSD1.0, AAD0.5 saved the energy by an addition 12%.

4.3. Experiment set II

DPM strategies try to adapt to changes in the network. It is important to demonstrate that a DPM method performs well in a lightly-loaded network, as well as in a congested one. In this set of experiments, we evaluate the performance of different PM strategies in a multi-client environment. There are 10 clients in the network, and each of them submits 1000 requests at a randomly generated think time interval between 1 and 3 s. The response time is according to the UC-Berkley-Home-IP trace. The Fig. 12a presents the RTT delay of the PM strategies in comparison to the baseline RTT time measured under the multiple client and NO_PM configuration. Fig. 12b plots the average NIC energy consumption results per request.

When there are multiple clients, the RTT of each request naturally increases due to collision. Simulation results show that the average baseline (NO_PM) RTT measured in a multiple client setting is 1.39 times higher than the average RTT measured in a single client scenario. For this reason, the relative RTT delays of different PM strategies actually decreased in a multi-client scenario. We observe that the performance of static PM methods, such as the PSM, does not change much in the multi-client setting because they do not react to changes in the environment. Conservative DPM strategies, such as FTDPM and BSD, react to collision by increasing the awake time for the NIC. Thus, we notice a significant reduction in RTT delay and a sharp increase in the NIC energy consumption at the same time. Predictive DPM approaches, like the AADPM, on the other hand, try to balance the energy consumption and throughput by adapting to the ever changing network condition. AADPM variations showed a moderate reduction in RTT delays and a small increase in the NIC energy consumption.

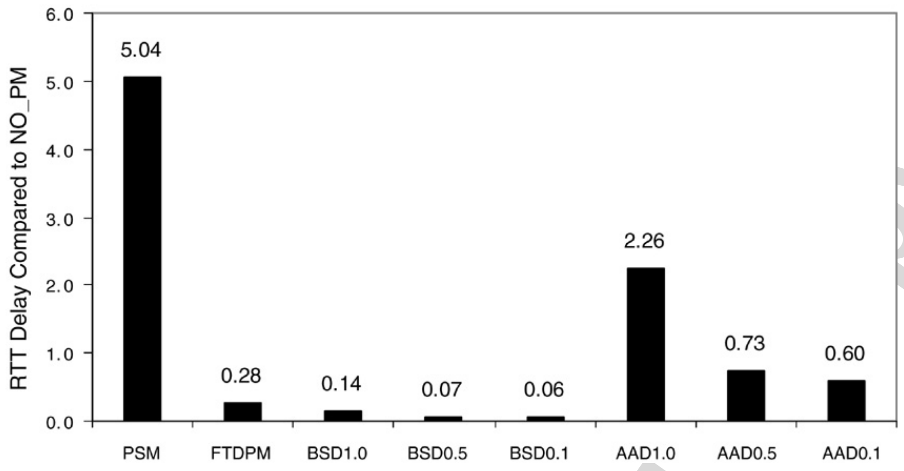


Fig. 12a. RTT delay ratio (multiple mobile stations).

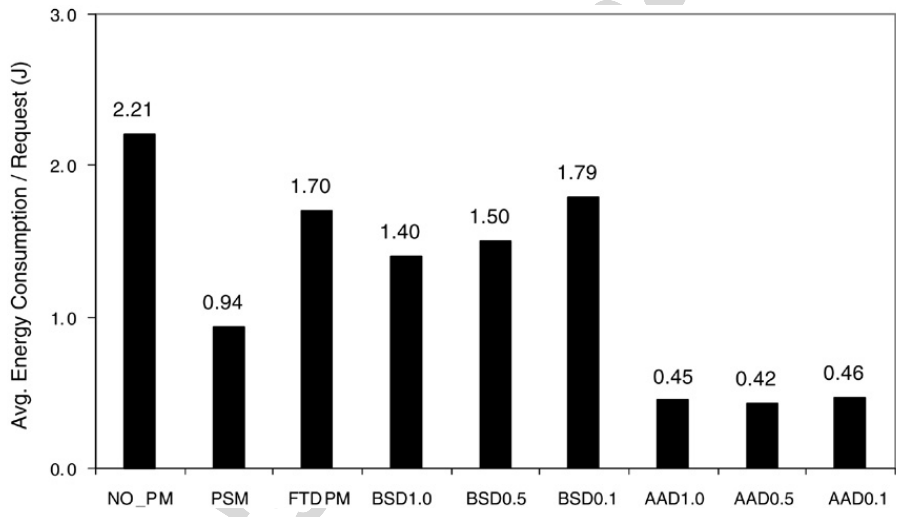


Fig. 12b. Average NIC energy consumption per request (multiple mobile stations).

Fig. 13 shows the average device energy consumption per request in the multi-client environment. It is interesting that dynamic PM strategies such as FTDPM and BSD0.1 cost more overall device energy to execute a request than the static PSM approach. The execution traces generated by our simulator suggest that this is mainly due to the NIC power management mode change overhead. Since FTDPM and BSD operate at the link layer and without application-level knowledge, they conservatively assume that each outgoing packet is a request and each incoming packet is a response. Thus, when the network is congested and the packet loss increases, these two methods leave the NIC awake most of the time. In addition, NIC power management mode changes become more costly in a congested network. As a result, the energy loss from the RTT delays exceeded the energy gain from the NIC.

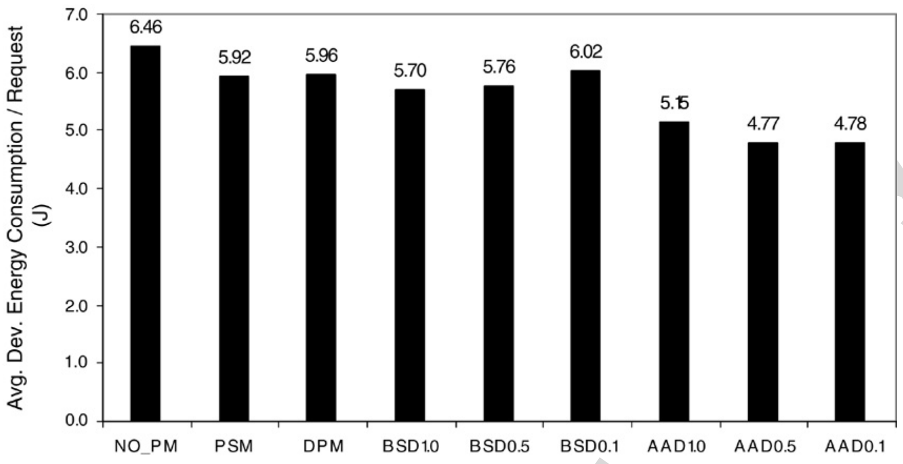


Fig. 13. Average device energy consumption per request (multiple mobile stations).

Table 2
Summary of notation

Notation	Meaning
P	The total power consumption of a mobile device
α	The average percentage of power consumed by the NIC with respect to the overall power consumption per request
$\alpha \times P$	The average NIC power consumption
$(1-\alpha) \times P$	The base power of a mobile device
T	Average RTT per request without PM
D	Average RTT delay when a PM strategy is applied
S	The percentage of average NIC energy saving compared to NO_PM

5. Discussion

Fig. 11 suggests that a power management scheme, e.g. PSM, may actually have adverse impact on the overall energy consumption. Observing this phenomenon, one may ask a question: is it possible that a PM strategy increases not only the RTT delay but also the overall mobile device energy consumption?

Using the notation summarized in Table 2, we can describe the expected device energy consumption of NO_PM as $T \times P$. A PM strategy can improve the overall energy saving of a mobile device if the inequality of (4) holds. In other words, if the inequality of (4) dose not hold, the PM strategy should not be employed because its performance is worse than NO_PM. Parameters P and α can be obtained from the hardware specifications, and parameters T, D, and S can be empirically determined by using training data. The simplified inequality shows that only S, D, and α are the final determinants for whether a power management scheme should be applied.

$$T \times P > (1+D) \times T \times [(1-\alpha) \times P + (1-S) \times \alpha \times P]$$
$$S > \frac{D}{\alpha \times (1+D)}. \quad (4)$$

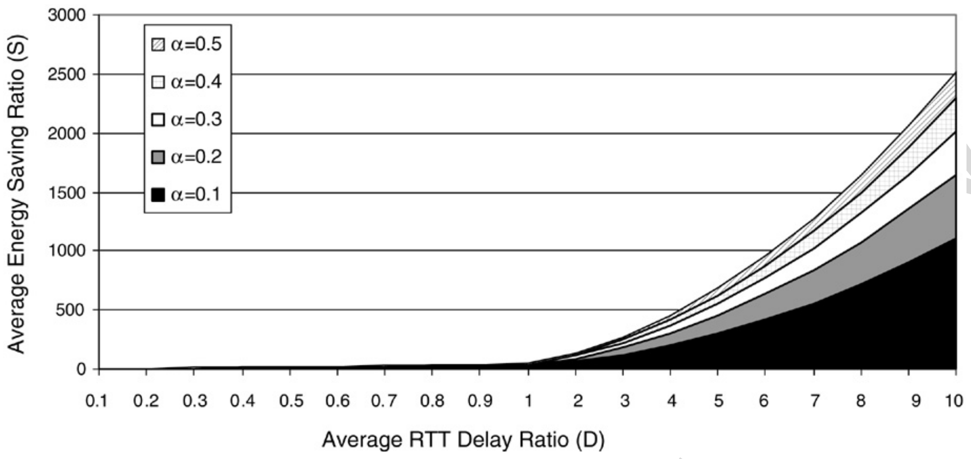


Fig. 14. Overall energy saving conditions.

Fig. 14 visualizes the inequality expressed by Eq. (4), with different α values. For example, when α equals to 0.2, the average NIC energy consumption constitutes 20% of the overall device energy consumption. If empirical studies show that the relationship between D and S falls in the grey or the black area, PM schemes should be avoided.

6. Conclusions

In this paper, we proposed an application-driven power management strategy, AADPM, which has online learning capability. Because it does not require changes to the existing IEEE802.11b protocol, it can be easily integrated with the majority of the request/response type wireless applications. Since AADPM does not assume a particular network usage pattern, it provides a general DPM solution for this type of applications.

Despite the fact that many PM strategies have been proposed, a study on the adaptability of different PM methods in the presence of multiple users is still missing. In this study, we compare the PM strategies under both the single-user and multiple-user scenarios. The experimental results show that AADPM gained the highest energy saving among all the PM strategies examined in all cases. When the server response time is longer than that for web browsing applications (several seconds long), AADPM achieved both the lowest energy usage and RTT increase. However, the results also indicate that AADPM is not as suitable for latency sensitive applications that have extremely short server response times, because it introduces longer RTT delays than the BSD and FTDPM methods in this situation.

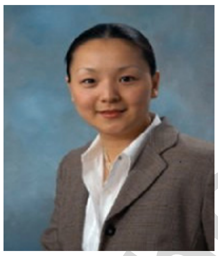
Static PM strategies like PSM do not adapt to the environmental changes and therefore, they are not noticeably affected by concurrent users. DPM approaches showed increases in both RTT delay and energy consumption when multiple users contend for the wireless media. Because of AADPM's adaptive learning capability, it achieved the lowest increase in RTT delay as well as energy consumption.

Acknowledgements

This work in part has been supported by The Office of Naval Research under contract N00014-02-1-0282 and National Science Foundation under contracts IIS-0324835 and STI-0129682.

References

[1] M. Anand, E.B. Nightingale, J. Flinn, Self-tuning wireless network power management, in: Proceedings of ACM MOBICOM, San Diego, CA, September 2003, pp. 176–189.
[2] L. Benini, A. Bogliolo, G.D. Micheli, A survey of design techniques for system-level dynamic power management, IEEE Transactions on VLSI Systems 8 (3) (2000) 299–316.
[3] Cisco Systems, Inc., Cisco Aironet 802.11a/b/g Wireless LAN Client Adapters Installation and Configuration Guide, 2001.
[4] IEEE Computer Society LAN MAN Standards Committee, IEEE Std 802.11: Wireless LAN Medium Access Control and Physical Layer Specifications, August 1999.
[5] S. Irani, S. Shukla, R. Gupta, Online strategies for dynamic power management in systems with multiple power-saving states, ACM Transactions on Embedded Computing Systems 2 (3) (2003) 325–346.
[6] Y. Jiao, A.R. Hurson, Application of mobile agents in mobile data access systems — a prototype, Journal of Database Management 15 (4) (2004) 1–24.
[7] R. Krashinsky, H. Balakrishnan, Minimizing energy for wireless web access with bounded slowdown, in: Proceedings of ACM MOBICOM, Atlanta, GA, July 2002, pp. 119–130.
[8] R. Kravets, P. Krishnan, Application-driven power management for mobile communication, ACM Wireless Networks 6 (4) (2000) 263–277.
[9] B.A. Mah, An empirical model of http network traffic, in: Proceedings of IEEE INFOCOM, Kobe, Japan, April 1997, pp. 592–600.
[10] E. Shih, P. Bahl, H. Balakrishnan, Wake on wireless: An event-driven energy saving strategy for battery operated devices, in: Proceedings of ACM MOBICOM, Atlanta, GA, July 2002, pp. 160–171.
[11] The VINT Project, The NS Manual, 2001.
[12] UC Berkeley Home IP Web Traces. Available from: http://ita.ee.lbl.gov/html/contrib/UCB.home-IP-HTTP.html.



Yu Jiao is a postdoctoral researcher at the Oak Ridge National Laboratory. She received the B.Sc. degree in computer science from the Civil Aviation Institute of China in 1997. She received her M.Sc. and Ph.D. degrees from The Pennsylvania State University in 2002 and 2005, respectively, both in computer science. Her main research interests include software agents, pervasive computing and secure global information system design.

A.R. Hurson is a Computer Science and Engineering Faculty at The Pennsylvania State University. His research for the past 25 years has been directed toward the design and analysis of general as well as special purpose computer architectures. His research has been supported by NSF, NCR Corp., DARPA, IBM, Lockheed Martin, ONR and Penn State University. He has published over 250 technical papers in areas including database systems, multidatabases, global information sharing processing, application of mobile agent technology, object oriented databases, mobile computing environment, computer architecture parallel and distributed processing. Dr Hurson served as the Guest Co-Editor of special issues of the *IEEE Proceedings on Supercomputing Technology*, the *Journal of Parallel and Distributed Computing on Load Balancing and Scheduling,* the *journal of integrated computer-aided engineering on multidatabase and interoperable systems, IEEE Transactions on Computers on Parallel Architectures and Compilation Techniques, Journal of Multimedia Tools and Applications, and*

Journal of Pervasive and Mobile Computing. He is the co-author of the *IEEE Tutorials on Parallel Architectures for Database Systems, Multidatabase systems: an advanced solution for global information sharing, Parallel architectures for data/knowledge base systems, and Scheduling and Load Balancing in Parallel and Distributed Systems*. He is also the guest Editor of advances in computers for Parallel, Distributed and Pervasive Computing.

He served as a member of the IEEE Computer Society Press Editorial Board, an IEEE Distinguished speaker, editor of IEEE transactions on computers, and IEEE/ACM Computer Sciences Accreditation Board. Currently, he is an ACM lecturer and editor of Journal of Pervasive and Mobile Computing.

Behrooz A. Shirazi is the Huie-Rogers Chair Professor and the Director of the School of Electrical Engineering and Computer Science at Washington State University. Prior to joining WSU in 2005 he was on the faculty of Computer Science and Engineering at the University of Texas at Arlington and served as the department chair from 1999 to 2005. Dr Shirazi has conducted research in the areas of pervasive computing, software tools, distributed real-time systems, and parallel and distributed systems over the past eighteen years. Dr Shirazi is currently serving as the Editor-in-Chief for Special Issues for Pervasive and Mobile Computing (PMC) Journal and has served on the editorial boards of the IEEE Transactions on Computers and Journal of Parallel and Distributed Computing in the past. He is a co-founder of the IEEE International Conference on Pervasive Computing and Communications (PerCom). He has served on the programme committee of many international conferences. He has received numerous teaching and research awards and has served as an IEEE Distinguished Visitor (1993–96) as well as an ACM Lecturer (1993–97).