CPM: Context-Aware Power Management in WLANs

Fahd Albinali and **Chris Gniady**
Computer Science Department
University of Arizona
Tucson, Arizona 85704
{albinali, gniady}@cs.arizona.edu

Abstract

In this paper, we present a novel approach for tuning power modes of wireless 802.11 interfaces. We use K-means and simple correlation techniques to analyze user's interaction with applications based on mouse clicks. This provides valuable contextual hints that are used to anticipate future network access patterns and intent of users. Based on those hints, we adapt the power mode of the wireless network interface to optimize both energy usage and bandwidth usage. Evaluation results (based on real data gathered from interaction with a desktop) show significant improvements over earlier power management schemes.

Introduction

Mobile devices are part of our life. We depend on them to provide good performance and long battery life. Reducing energy consumption and therefore prolonging battery life has become one of the most important challenges in designing future computing systems. While Moore's Law provides steady reduction in power consumption per operation, increasing demand for higher performance, versatile functionalities, and better user interfaces has been raising power consumption faster than the reduction from semiconductor technology.

Performance and energy consumption are tightly coupled where higher performance is usually achieved at the cost of increased power. Higher power does not necessarily translate into increase in energy consumption. For instance, hardware in a higher performance state usually accomplish a particular task faster than hardware in lower performance states. This reduces the time during which the entire system has to be on. However, tasks that demand high performance are usually interleaved with tasks that do not have specific performance demands. Keeping the device in one performance level is inadequate, as it may waste valuable energy resources. The challenge in designing mobile systems therefore lies in providing appropriate performance levels that closely match the needs of different tasks.

In this paper, we focus on reducing energy consumption in wireless Network Interface Cards (NICs) while maintaining good performance levels. IEEE 802.11 Wireless NICs usually offer three power states: Continuous Aware Mode (CAM), Power Saving Mode (PSM), and the Off State. The off state can be used only when no communication applications are running. Once a communication application is running, we can use CAM or PSM to provide two different bandwidths and power consumption levels depending on application requirements. Ideally, the goal is to switch to CAM during high bandwidth demand and to PSM otherwise. The critical issue is to predict when the appropriate mode is needed so we may provide high performance while minimizing energy consumption.

To address the above, we propose Context-aware Power Management (CPM): a framework that exploits user interaction with an application (particularly mouse clicks) to control the power modes of wireless interfaces. We incorporate high-level contextual information about user's activity (e.g. chatting, using webcam) to anticipate network access patterns. The idea is motivated by observing that network traffic (for interactive applications) usually follows after a specific interaction with the application interface. For instance, when the user clicks the webcam button, it is reasonable to expect additional network traffic. CPM observes mouse clicks that lead to specific network activity and correlates such interactions with resulting network activity. Once similar patterns of mouse clicks are observed, the network behavior is predicted and the network card is switched to an appropriate power mode.

Compared with previously proposed state-of-the-art Self-Tuning Power Management (STPM)(Anand *et al.* 2003), CPM has two major advantages. First, it correlates network activity to actual user behavior. Second, it is more deployable in existing systems as it does not require application modifications. These advantages provide a robust predictor that adapts to changing user behavior and maximizes energy savings. We evaluated CPM using a detailed trace-driven simulation. Our results show that CPM is more adaptive to user's bandwidth demands and provides higher-energy savings than STPM. More importantly, our results can be easily extended to general resource configuration problems. Subsequently, this paper makes the following contributions:

- We are first to exploit user interactions with an application for energy management.
- We propose a complete design framework for implement-

ing CPM in a computer system.

- We evaluate and demonstrate the effectiveness of CPM in managing wireless NICs.

The rest of the paper is organized as follows. Section 2 reviews current energy saving techniques in wireless interfaces. Section 3 presents the design of CPM. In Section 4 and Section 5, we present our experimental setup and evaluation of CPM. Section 6 presents related work and finally, Section 7 draws concluding remarks.

Motivation

The IEEE 802.11 standard (IEEE 1999a) offers two power management schemes: Continuous Aware Mode (CAM) and Power Saving Mode (PSM). The default operational mode is CAM where wireless interfaces continuously listen to the shared physical medium. This enables NICs to respond to user's requests immediately thereby minimizing delay while maximizing throughput. Alternatively, PSM periodically wakes up the network interface switching it to a high power state during which transmissions can occur. Once transmissions complete, the network interface goes back to sleep. This significantly reduces power consumption (IEEE 2001).

Ideally, power management schemes should minimize energy consumption without sacrificing performance. To achieve this, the network card should be switched immediately to CAM when a high bandwidth transfer occurs and back to PSM when the transfer is over. It is important to emphasize that keeping the card in PSM all the time may increase energy consumption of the entire system. Because PSM operates at a lower data rate, PSM transmission delays are longer which often keeps the rest of the system operating for longer periods. Clearly, this may result in higher overall energy consumption in the system. Moreover, it has been shown that PSM can cause unacceptable 16-32x slowdown for synchronous traffic (Anand *et al.* 2003).

The above observations motivated the development of adaptive power management schemes (Anand *et al.* 2003) that switch dynamically between CAM and PSM to balance performance with power consumption (Anand *et al.* 2003). The design of STPM is based on five requirements:

1. Applications issue explicit hints about future network activity.
2. The application hints are issued ahead of time so that the network power mode is changed to the proper state before network packets arrive.
3. The application hints state if network transfers are latency sensitive or not, so that power management can be tailored for maximum energy savings or performance.
4. Users specify a desired performance/energy level through a tunable parameter.
5. Finally, the entire system energy has to be considered to minimize the overall energy consumption of the system rather than that of the wireless interface.

We observe that all of the above aspects require some interaction with the end-user or the programmer. Requirements 1-3 require the programmer to modify the application in order to provide hints about network traffic. Requirements 4 and 5 are STPM parameters that are configured by the user to set the desired levels of energy and performance for the system.

STPM can be very accurate, as the programmer knows what part of the applications is executing and perhaps what kind of behavior can be expected from the user. However, this introduces power management as another optimization dimension to already difficult programming requirements that target among others performance, reliability, and usability. Moreover, the underlying hardware is usually not known during application design, therefore it may be difficult (or even impossible) to generate hints that take into account differences in hardware. This approach is also less adaptive to changing hardware. For example, hardware designers may introduce additional power modes or parameters, hinting applications in STPM may require modifications to take advantage of new features.

To support unmodified legacy applications, STPM profiles network traffic to anticipate future network activity. Passively monitoring low-level network access provides little information about user's current context (e.g. is the user chatting or using her webcam). In this case, the context of execution of an application is completely lost and heuristic profiling is not able to fully realize the potential of STPM. We emphasize that the context of execution of an application captures both user's and application's intent which are critical for accurately estimating future network activity. Not exploiting this information reduces the overall accuracy of predicting future access patterns that often leads to suboptimal switching between CAM and PSM. This impacts both performance and energy savings in STPM.

In CPM, we explore an alternative way of providing context for managing power in network interfaces. CPM transparently monitors user interactions and automatically generates hints for NIC power management. In our approach, user interaction is correlated to resulting network activity, since the user is directly responsible for I/O traffic in interactive applications. This provides rich contextual cues with the following benefits:

- Without modifying applications or system components, interactions that require higher bandwidths (e.g. streaming video) can be easily inferred.
- Hints are provided ahead of actual transfers since user interacts with applications before transfers occur.
- Both foreground and background transmissions can be inferred since foreground transmissions will usually come from foreground windows.

The remaining requirements 4 and 5 from STPM remain unchanged and can be directly applied in CPM design. Therefore, CPM can potentially take full advantage of STPM-like mechanisms and provide accurate and timely predictions in unmodified legacy applications.

Context-Aware Power Management

Design of CPM has to satisfy four requirements. First, hints (that anticipate network activity) should be generated automatically and transparently to the user. Second, hints should

be timely and accurate so that power mode switching for NICs occurs right before high or low activity periods. Third, collecting inputs and training should be done online to capture user behavior as it occurs. Finally, both training and prediction techniques that accomplish timeliness and accuracy have to be relatively simple to reduce computational overhead.

Monitoring User Behavior

Users in GUI environments typically control their applications using mouse clicks. The simple point-and-click style has been popularized by the World Wide Web (WWW), where navigation is predominantly click-based through hyper-links. We observe that in current multimedia systems and web browsers, virtually all interactions can be accomplished through mouse clicks (Dix *et al.* 2003). This trend in interface design, coupled with a tremendous increase in web-enabled applications, adds to the correspondence between user interaction (with applications) and underlying network activity. For example, as soon as a user clicks in a URL box in a web browser and starts typing, it is reasonable to assume that an HTTP/FTP request will follow resulting in additional network traffic. We therefore argue that correlating GUI interactions that invoke network aware features to actual network activity will allow systems to make better assessments of future network access patterns.

CPM monitors mouse interactions for all network aware applications and leverages this information to infer network activity. Monitoring starts, only after a network activity is detected in the application to prevent unnecessary monitoring of applications that do not use the network. Once monitoring starts, CPM records and stores mouse interactions for different application windows. CPM then clusters these interactions and correlates each cluster with underlying network activity. This simple mechanism identifies clusters of mouse interactions that may result in substantial network activity. This in turn is used to issue hints that anticipate network traffic.

It is important to note that monitoring mouse clicks enables CPM to be timely– it properly adjusts the power state of the NIC before the corresponding network activity occurs. Timeliness in CPM comes from user think-time. For example, when a user clicks on a file transfer menu and spends time selecting a file, the associated delay provides valuable time for CPM to process captured clicks, generate hints, look up resulting network activity in prediction tables, and switch to a suitable power state.

Finally, we emphasize that our approach can be extended to include additional inputs (such as keyboard events) particularly when mouse interactions are insufficient in characterizing unique contexts. We do not purse additional forms of inputs in this paper, since mouse clicks provided sufficient context for accurate prediction of network activity in the applications we evaluated. We anticipate exploring additional forms of input in our future work.

Mapping Mouse Clicks to Hints

The main goal of CPM is to generate two types of hints based on mouse inputs: (1) *CAM Hints:* for high bandwidth delay sensitive activities (e.g. audio streaming) that recommend switching to CAM and (2) *PSM Hints:* for low bandwidth activities (e.g. text messaging) that recommend switching to PSM. The key challenge in inferring these hints is how to move from low-level signals (e.g. such as mouse clicks or keyboard events) to meaningful high-level contexts such as user is streaming video and therefore requires higher bandwidth.

In interactive applications, interface components are usually bound to one particular function (e.g. the webcam button starts webcam). For a given window, mouse clicks that correspond to specific functions will appear in clusters. Figure 1 shows an interaction scenario with one application window and four clusters of mouse clicks. There is a clear and direct correspondence between the clusters and functional regions of the application interface. For example, cluster 1 is responsible for user initiating file transfers or streaming through webcam and cluster 2 corresponds to user clicking on the text box before typing a text message. This correspondence will remain static as long as the user does not change the layout of the interface. Such changes are infrequent for two reasons. First, some applications do not allow interface modifications. Second, once a user modifies the interface to her liking, the interface remains static.

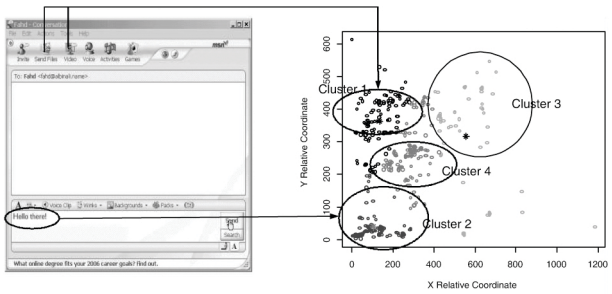


Figure 1: Example Interaction Scenario

CPM differentiates functional clusters of mouse clicks using a simple clustering technique, namely K-means. As illustrated in Figure 1, this has the desirable effect of identifying classes of mouse clicks that correspond to mouse-driven functions of the software. CPM then correlates each cluster with underlying network activity using Pearson's Correlation. Clusters that correlate with periods of high network activity are labeled *high activity clusters*, otherwise they are labeled *low activity clusters*.

Once clustering converges, CPM classifies new mouse clicks with respect to the identified clusters and issues hints appropriately: if the mouse click belongs to *high activity clusters*, CPM switches the NIC to CAM. If the mouse click belongs to *low activity clusters*, CPM switches the NIC back to PSM provided there are no on-going high activity transmissions.

Proposed CPM Architecture

CPM runs as a system-level daemon that consists of three components as shown in Figure 2. *The Application Agent:* intercepts specific mouse events for a host of applications

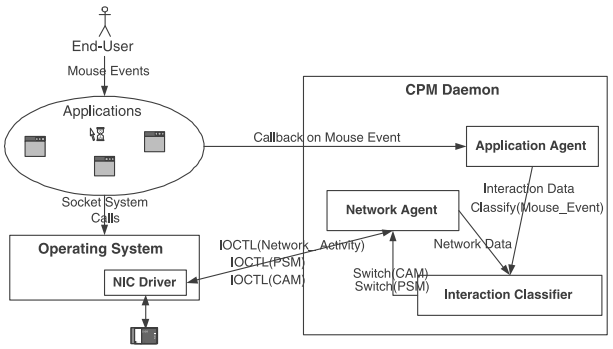


Figure 2: System Architecture

and records the events on a per window basis. Each interaction is recorded with a global time stamp. *The Network Agent:* is responsible for gathering and recording statistical data about network activity that is used by the classifier. The Network Agent also controls the power mode of the wireless NIC. *The Interaction Classifier:* is responsible for clustering user clicks (using K-means) and correlating them to underlying network activity (using Pearson's correlation). The Interaction Classifier issues hints for the Network Agent with recommendations to switch to CAM or PSM depending on the classification of mouse inputs. We emphasize that the use of an independent daemon design reduces required modifications to the operating system and improves deployability.

Methodology

To accurately compare CPM with STPM we use trace base simulation. We have implemented STPM simulator according to the authors' specifications (Anand *et al.* 2003). To implement CPM we modified the module in our implementation of STPM that is responsible for receiving hints from the application to receive similar hints from the CPM daemon. Since we did not have applications modified with the hints for STPM simulation, we leveraged the benefit of trace based simulation and looked into future events in the trace and generated perfect hints for STPM.

We decided to test our system in the context of an Internet messenger application namely Microsoft Messenger (MSN). Our choice was motivated by the following two reasons. First, MSN Messenger is a real-world application that hosts a variety of functions including messaging, file transfer, streaming audio or video. Second, MSN generates different traffic patterns depending on the functions used by the end-user (e.g. short bursts for sending messages, longer bursts for sending files or streaming), therefore it thoroughly tests our mechanism.

Evaluation

Our evaluation serves as a proof of concept that interaction provides valuable context that can be used to configure underlying system resource, in this case, the NIC to reduce energy consumption. We collected user interaction with MSN for a period of 26 hours. The period is long enough to show different usage patterns for work related and leisure activities. In the trace, the user used three features of MSN Messenger including: messaging, webcam and file transfer. We partitioned our data into two groups: (a) training data with 621 mouse events that correspond to the first 21 hours and (b) test data with 181 mouse events that correspond to the last 5 hours.

Table 1: Performance for Different Power Modes

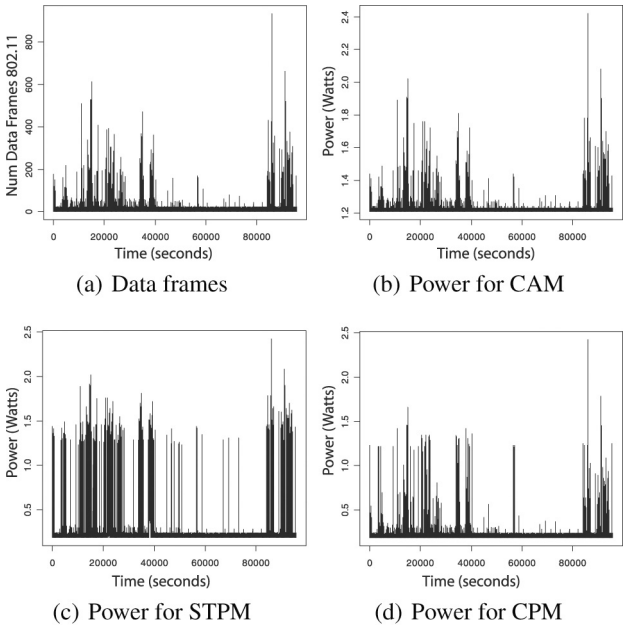
Power Mode	No. Changes to PSM/CAM	Total Energy (Joules)	Pearson's Correlation
CAM	0	117188	0.99
PSM	0	20946	0.99
STPM	915	26047	0.72
CPM	984	24619	0.89

To identify interactions that map to mouse-driven functions of MSN, we used K-means to find a set of centers that reflect the distribution of mouse clicks for each MSN window. K-means partitions the data into a number of clusters that maximize the separation between clusters (i.e. inter-cluster space) while minimizing the compactness of each cluster (i.e. intra-cluster space). We ran the algorithm repetitively using different numbers of clusters to identify optimal partitioning of the data set. Our optimality criteria implies that K-means partitions the data into clusters that correspond to actual interaction centers on an interface as shown earlier in Figure 1. We choose the value for K that produces hints that maximize energy savings for our training data. For our particular data set, the best K value was 4.

Figure 1 shows the results of K-means clustering and the centers for each cluster. As we anticipated, the clusters corresponded to the layout of MSN's interface. Cluster 1 in the upper left corner corresponds to both file transfer and web cam interactions (since both buttons are adjacent and consume high bandwidth) and cluster 2 in the lower left corner corresponds to sending messages through clicking in the MSN Messenger text box. Cluster 3 and 4 correspond to mouse interactions that did not result in network activity (e.g. dragging the window). CPM labeled cluster 1 with a high activity label.

Accuracy of cluster detection and correlation in CPM translates into higher energy savings than in STPM, as shown in Table 1. CAM results in the highest energy costs of approximately 117 KJoules. PSM represents a lower bound on energy consumption but suffers from excessive delays due to dozing between successive beacons. Both STPM and CPM (our proposed algorithm) represent a compromise where the system switches dynamically between CAM and PSM to balance energy consumption with bandwidth performance. Energy savings in CPM come from two sources: higher switching activity between CAM and PSM and better timeliness and switching accuracy.

CPM has 69 additional switches from CAM to PSM on top of 915 switches issued by STPM. It is important to emphasize that additional switching from CAM to PSM will always reduce power consumption. However, the key thing



(a) Data frames (b) Power for CAM

(c) Power for STPM (d) Power for CPM

Figure 3: Frame and Power Distributions for MSN Data

here is that the switching to CAM should correspond to high traffic periods and switching to PSM should correspond to low traffic periods. Dynamic mechanisms that consistently switch to CAM during high traffic and back to PSM during low traffic will show high correlation between the distributions of transmitted data frames and consumed power levels. On the other hand, static mechanisms that use one particular power mode (e.g. CAM) will show an almost 1.0 correlation between the data distribution and the power-levels consumed but at higher baseline power levels (see Figure 3(b)).

Figure 3 shows empirical distributions for our MSN test data and corresponding power levels for different operational modes. Figure 3(a) represents the distribution of the received and sent data frames. Ideally, a dynamic power management algorithm should stay in PSM (i.e. low power mode) and switch to CAM only when there is significantly large network activity to minimize network delay.

Figure 3(b) shows that CAM closely matches the data distribution thus responding to large transfers almost immediately. However, CAM baseline power (or minimum power consumption) is 1.21 Watts which results in overall energy consumption of 117 KJoules.

STPM in Figure 3(c) takes advantage of PSM when no data is being transferred thus its baseline power is 0.19 Watts. This significantly reduces the energy consumption to 26 KJoules. STPM however results in high mispredictions reflected in coarser graph regions and the mismatch between the power distribution and the data distribution. This observation is verified by measuring Pearson's correlation between power and data distributions which is 0.72 (see Table 1).

Finally, Figure 3(d) shows the effect of adding CPM contextual cues to STPM. The overall energy consumption is further reduced to 24.5 KJoules. The power distribution is more adaptive to variations of the data distribution with a correlation of 0.89. This suggests that high-level contextual cues can improve the prediction of network access patterns and perhaps more generally prediction of device access patterns.

Related Work

Perceptual information can be leveraged to customize systems to end-user's situation or context. Context includes many aspects of users environment such as location, user activities and system-user interaction. One of the earliest efforts in this area was the Lumiere Project (Horvitz *et al.* 1998). Lumiere served as the basis for Microsoft's Office Assistant. The project used Bayesian models to reason about the time-varying goals of computer users from observed interaction and queries. More specifically, Lumiere intercepted interaction events and mapped them into observation variables in a Bayesian network. This network was then used to provide assistance to end-users. In CPM, we extend systems like Lumiere to adapt and customize system-level resources such as power.

The Location Stack project (LaMarca *et al.* 2005) used location based contextual hints to support a range of location-sensitive services. Unlike CPM, the Location Stack focused on information retrieval and customization. More recently, event-based mechanisms were used for building environments that respond to users activities (Albinali *et al.* 2003). Using ubiquitously deployed sensors, perceptual systems in smart environments monitored user behavior and issued hints about user's activities that customized devices. This approach focused on customizing displays and calibrating intelligent artifacts. Finally, layered hidden Markov models (Oliver *et al.* 2004) were used to characterize states of a user's activity based on streams of video, audio and computer (keyboard and mouse) interactions. This approach focused on daily user activities (e.g. phone conversation, doing a presentation) and was never applied to resource allocation and customization problems.

Earlier efforts have also addressed power management issues in WLANs. STPM explored the use of explicit application-level hints and on-line modeling of application access patterns to set network power management parameters (Anand *et al.* 2003). Application controlled power management has been applied to other devices and system components (Ellis 1999; Heath *et al.* 2002; Lu *et al.* 2000; Weissel *et al.* 2002) with high potential for reducing energy consumption. However, this technique places a burden of inserting power management directives on the programmers and requires existing applications to be modified before they can benefit from energy management. To address those issues, mechanisms for automatic application-like hint generation were proposed (Gniady *et al.* 2004). Finally, transport-level mechanisms have also been proposed to optimize energy usage based on passive monitoring of low-level network behavior (Kravets *et al.* 2000). Unlike CPM, these approaches require modifying several system components and depend on low-level network behavior that does not necessarily characterize future needs.

Conclusion

In this paper, we make two contributions: (1) our approach is the first power management scheme that tunes the power characteristics of the network interface based on user's interaction with running applications. Because user interaction with a desktop is generally purposeful and intentional, interaction histories provide good contextual cues about future device access patterns. We demonstrate the effectiveness of this approach in the context of wireless networks where the power of wireless interfaces is driven by a history of mouse clicks. (2) We provide evidence that suggests that simple machine learning techniques such as K-means coupled with correlation of clusters with observed device activity can be used to distinguish different interaction patterns.

We showed that explicit monitoring of user interaction with applications is an important component in hint generation; instead of monitoring low-level application behavior (e.g. network activity), as current systems do, we explicitly bring user interaction as a key player. We suggest that this approach will both improve accuracy of hint generation and will respond to new demands of emerging ubiquitous applications in ways that today's hint generation mechanisms cannot.

Exploiting user interaction as a high-level behavioral cue can potentially bridge the gap between user's high-level actions and underlying resource allocation/configuration strategies. The approach has the following benefits:

1. *Proactive Behavior:* Rather than having systems that react to changes as they occur, proactive behavior enables systems to sense and anticipate contextual changes and spontaneously allocate/configure resources to meet future demands. CPM allows for proactive behavior in two ways: (1) automated generation of hints without user involvement and (2) timely generation of hints before resources are needed.

2. *Accuracy:* Existing hint generation mechanisms typically analyze low-level data streams (e.g. network packets) to anticipate future resource requirements. This implicitly assumes that low-level data patterns can sufficiently determine future resource requirements and ignores the direct and compelling relationship between user interaction and resource requirements. CPM exploits this relationship as a key determinant of future needs.

3. *Low Overhead:* CPM selectively and transparently intercepts user interaction with targeted applications. This minimizes the overhead of using CPM with respect to both performance and programmability: applications may choose not to use CPM (e.g. when resources are not in contention) and programmers are not required to modify their applications. Moreover, CPM classifies input based on the Euclidean distance with respect to different clusters. This approach has low overhead with time complexity O(n) where n is the number of clusters.

4. *Deployability:* Perhaps most importantly, CPM is easily and rapidly deployable in the context of existing applications. CPM can be used with other optimization schemes such as STPM, which can further enhance the performance of systems.

Currently, we are looking at potential interactions between CPM and other similar schemes. Our plans for future work include exploiting more elaborate techniques for recognizing complex interaction patterns that for example may include a sequence of mouse clicks and keyboard events. Finally, we plan to explore further the notion of interaction-based power management in the context of disk usage.

References

Albinali, F.; Boddupalli, P.; Davies, N.; and Friday, A. 2003. Correlating sensors and activities in an intelligent environment: A logistic regression approach. 2875:318–333.

Anand, M.; Nightingale, E. B.; and Flinn, J. 2003. Self-tuning wireless network power management. 176–189.

Dix, A.; Finley, J.; Abowd, G.; and Beale, R. 2003. *Human-computer interaction (3rd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Ellis, C. S. 1999. The Case for Higher-Level Power Management. In *Workshop on Hot Topics in Operating Systems*, 162–167.

Gniady, C.; Hu, Y. C.; ; and Lu, Y.-H. 2004. Program counter based techniques for dynamic power management. In *Proceedings of the 6th Symposium on Operating Systems Design and Implementation*.

Heath, T.; Pinheiro, E.; Hom, J.; Kremer, U.; and Bianchini, R. 2002. Application transformations for energy and performance-aware device management. In *Proceedings of the 11th International Conference on Parallel Architectures and Compilation Techniques*.

Horvitz, E.; Breese, J.; Heckerman, D.; Hovel, D.; and Rommelse, K. 1998. The lumiere project: Bayesian user modeling for inferring the goals and needs of software users. *the Fourteenth Conference on Uncertainty in Artificial Intelligence*.

IEEE. 1999a. 802.11 wireless standard. Online Technical Standard Specification. http://grouper.ieee.org/groups/802/11/.

IEEE. 1999b. Management information base. Online Technical Standard Specification. http://grouper.ieee.org/groups/802/11/.

IEEE. 2001. Supplement to ieee standard for information technology: Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. Online Technical Standard Specification. http://grouper.ieee.org/groups/802/11/.

Kravets, R., and Krishnan, P. 2000. Application-driven power management for mobile communication. *Wirel. Netw.* 6(4):263–277.

LaMarca, A.; Hightower, J.; Smith, I.; and Consolvo, S. 2005. Self-mapping in 802.11 location systems. 87–104.

Lu, Y.-H.; Micheli, G. D.; and Benini, L. 2000. Requester-aware power reduction. In *Proceedings of the International Symposium on System Synthesis*, 18–24.

Oliver, N.; Garg, A.; and Horvitz, E. 2004. Layered representations for learning and inferring office activity from multiple sensory channels. *Comput. Vis. Image Underst.* 96(2):163–180.

Weissel, A.; Beutel, B.; and Bellosa, F. 2002. Cooperative I/O—a novel I/O semantics for energy-aware applications. In *Proceedings of the Fifth Symposium on Operating System Design and Implementation*.