Application-specific network management for energy-aware streaming of popular multimedia formats

Surendar Chandra
University of Georgia
Athens, GA 30602-7404
surendar@cs.uga.edu

Amin Vahdat
Duke University
Durham, NC 27709-0129
vahdat@cs.duke.edu

Abstract

The typical duration of multimedia streams makes wireless network interface (WNIC) energy consumption a particularly acute problem for mobile clients. In this work, we explore ways to transmit data packets in a predictable fashion; allowing the clients to transition the WNIC to a lower power consuming *sleep* state. First, we show the limitations of IEEE 802.11 power saving mode for isochronous multimedia streams. Without an understanding of the stream requirements, they do not offer any energy savings for multimedia streams over 56 kbps. The potential energy savings is also affected by multiple clients sharing the same access point. On the other hand, an application-specific server side traffic shaping mechanism can offer good energy saving for all the stream formats without any data loss. We show that the mechanism can save up to 83% of the energy required for receiving data. The technique offers similar savings for multiple clients sharing the same wireless access point. For high fidelity streams, media players react to these added delays by lowering the stream fidelity. We propose that future media players should offer configurable settings for recognizing such energy-aware packet delay mechanisms.

1 Introduction

The proliferation of inexpensive, multimedia capable mobile devices and ubiquitous high-speed network technologies to deliver multimedia objects is fueling the demand for mobile streaming multimedia. Public venues [31] are deploying high speed IEEE 802.11b [24] based public wireless LAN networks. Commodity PDA devices that allow the users to consume mobile streaming multimedia are becoming popular. A necessary feature for mass acceptance of a streaming multimedia device is acceptable battery life. Advances in hardware and software technologies have not been matched by corresponding improvements in battery technologies. Future trends in battery technologies alone (along with the continual pressure for further device miniaturization) do not promise dramatic improvements that will make this issue disappear.

Newer hardware improvements are reducing the power consumption of system components such as back-lit displays, CPUs etc. However, WNICs operating at the same frequency band and range continue to consume significant power. Earlier work by Stemm et al. [32] reported that the network interfaces draw significant amounts of power. For example, a 2.4 GHz Wavelan DSSS card (11 Mbps) consumes 177 mW while in *sleep* state, but consumes 1319 mW while *idle*. Havinga et al. [18] noted that this Wavelan card consumes 1425 mW for receiving data and 1675 mW for transmitting data. For comparison, a fully operational Compaq iPAQ PDA only consumes 929 mW while the same iPAQ consumes 470 mW with the backlight turned off [14, 8]. For reference, the iPAQ is equipped with two 2850 mWh batteries, one each in the unit and the PCMCIA sleeve. Streaming media tends to be large and long running and consume significant amounts of network resources to download data. Hence, it is important to look at techniques to reduce the energy consumed by the network interface to download the multimedia stream.

Traditionally, reducing the fidelity of the stream and hence the size is a popular technique that is used to customize the multimedia stream for a low bandwidth network. Reducing multimedia fidelity can also be expected to reduce the amount of data and hence the total energy consumed. However, if care is not taken to return the network interface to the *sleep* state as much as possible, reducing the amount of transmitted data will have negligible effect on the overall client energy consumption. Frequent switching to low power consumption states also promises the added benefit of allowing the batteries to recover, exploiting the battery recovery effect [7].

In our earlier work [5], we explored the client WNIC energy implications of popular streaming formats (Microsoft media [28], Real media [29] and Quicktime [2]) under varying network conditions. We believe that these widely popular formats are more likely to be deployed than custom streaming formats (that are specially optimized for lower energy consumption). Based on our observations, we developed history based client-side techniques to exploit the stream behavior and lower the energy required to receive these streams. We illustrated the limitations of such client-side policies in predicting the next packet arrival times. These client-only policies do not allow us to achieve the potential energy saving for consuming multimedia streams without losing data packets.

In general, mechanisms that make the data packets arrive at predictable intervals can facilitate such transitions to lower power states. The choice of these transmission periods is a trade-off between frequent transitions to high power states and added delays in the multimedia stream reception. Such traffic shaping can be realized either in the origin server, in the network infrastructure closer to the mobile client and in the access point itself; at the MAC level. In this work, we analyze the effectiveness of these different approaches in regulating the streams to transmit data packets at regular, predictable intervals. Such packet arrivals enable client-side mechanisms to effectively transition the wireless interfaces to a lower power consuming *sleep* state.

In this work, we show the limitations of MAC level IEEE 802.11 power saving mode for isochronous multimedia streams. The access points have to balance potential energy savings for a single mobile client with a need for fair allocation of network resources. Without an understanding of the stream requirements, these MAC level mechanisms do not offer any energy savings for multimedia streams over 56 kbps. The potential energy savings also reduces for multiple clients sharing the same access point. On the other hand, a server side traffic shaping mechanism can offer good energy saving for all the stream formats without any data loss. We show that the mechanism can save up to 83% of the energy required for receiving useful data. The technique can also offer similar savings for multiple clients sharing the same wireless access point. For high fidelity streams, typical media systems react to these added delays by lowering the stream quality. We propose that future media players offer configurable settings for clients operating under energy conserving WLAN systems such that these delays are not associated with network congestion.

The remainder of this paper is organized as follows: Section 2 reviews our previous work as the necessary background and places our work in context to other related work. Next we present the experimental setup, evaluation methodologies, measurement metrics and the workloads used in our study in Section 3. Section 4 analyzes the effectiveness of IEEE 802.11 power management scheme to conserve energy for our streams. Section 5 explores the effectiveness of server side assistance in conserving the WNIC energy on the client. We conclude in Section 6.

2 Related work

There has been considerable work on power management for components of a mobile device. This work includes spindown policies for disks and alternatives [35, 3, 26, 12, 19], scheduling policies for reducing CPU energy consumption [34, 17] and managing wireless communications [20, 11, 21, 30]. Our work is similar in spirit to the work of Feeney et al. [15]. They obtain detailed measurements of the energy consumption of an IEEE 802.11 wireless network interface operating in an ad hoc networking environment. They showed that the energy consumption of an IEEE 802.11 wireless interface has a complex range of behavior and that the energy consumption was not synonymous with bandwidth utilization. Our work explores similar techniques for WNIC energy management for multimedia traffic.

Lorch et al. [27] presented a survey of the various software techniques for energy management. Havinga et al. [18] presented an overview of techniques for energy management of multimedia streams. Agrawal et al. [1] described techniques for processing video data for transmission under low battery power conditions. Corner et al. [10] described the time scales of adaptation for mobile wireless video-conferencing systems.

Ellis [13] advocates high level mechanisms for power management. Flinn et al. [16] demonstrated such a collaborative relationship between the operating system and application to meet user-specified goals for battery duration. Vahdat et al. [33] proposed that energy as a resource should be managed by the operating system. Kravets et al. [22] advocated an end-to-end model for conserving energy for wireless communications. In our earlier work [6], we utilized transcoding as an application level technique to reduce the image data; trading off image size for network transmission and storage costs. In this work, we explore high level mechanisms to enable the mobile client to transition to lower power states and reduce overall energy requirements.

2.1 Client-side history based adaptation mechanism

In our earlier work [5], we explored the energy implications of popular streaming formats (Microsoft media [28], Real media [29] and Quicktime [2]) under varying network conditions. We believe that these widely popular formats are more likely to be deployed than custom streaming formats (that are specially optimized for lower energy consumption). We showed that Microsoft media tended to transmit packets at regular intervals. For high bandwidth streams, Microsoft media exploits network level fragmentation, which can lead to excessive packet loss (and wasted energy) in a lossy network. Real stream packets tend to be sent closer to each other, especially at higher bandwidths. Quicktime packets sometimes arrive in quick succession; most likely an application level fragmentation mechanism.

Based on our observations, we developed history based client-side techniques to exploit the stream behavior and lower the energy required to receive these streams. We illustrated the limitations of such client-side policies in predicting the next packet arrival times. We showed that the regularity of Microsoft media packet arrival rates allow simple, history based client-side policies to transition to lower power states with minimal data loss. A Microsoft media stream optimized for 28.8 Kbps can save over 80% in energy consumption with 2% data loss. A high bandwidth stream (768 Kbps) can still save 57% in energy consumption with less than 0.3% data loss. For comparison, the WNIC was only receiving data for 0.45% and 14.51% of the time for these two streams, respectively. Also, both Real and Quicktime packets were harder to predict at the client-side without understanding the semantics of the packets themselves. Quicktime's energy savings came at a high data loss, while Real offered negligible energy savings. We believe that modifying Real and Quicktime services to transmit larger data packets at regular intervals can offer better energy consumption characteristics with minimal latency and jitter.

2.2 MAC level power saving modes

Wireless technologies such as IEEE 802.11 [25, 24] and Bluetooth [4] use a scheduled rendezvous mechanism of power saving wherein the wireless nodes switch to a low power *sleep* mode and periodically awaken to receive data from other nodes. Different wireless technologies utilize variations of this scheduled rendezvous mechanism. For example, IEEE 802.11 uses scheduled beacons along with a TIM/DTIM packet notification mechanism. Bluetooth, a wireless cable replacement technology, defines three different power saving modes; the *sniff* mode which defines a variable slave specific activity delay interval, the *hold* mode wherein the slave can sleep for a predetermined interval without participating in the data traffic and the *park* mode wherein the slave gives up its active member address. The potential energy saving progressively decreases from *sniff* to *hold* to *park* modes. In general, Bluetooth enabled devices with their range less than 10 meters consume less power than IEEE 802.11 based WLAN devices. For example, a typical Bluetooth device (Ericsson PBA 313 01/2) consumes 84 mW to receive data, 126 mW to transmit data while consuming as little as 2 mW in the *park* state. Energy consumption in a fully packaged system is likely to be higher. Bluetooth technology offers a vertically integrated solution with a lower data rate and range as compared to wireless LAN technologies. As such, Bluetooth technologies are tuned towards cable replacement rather than for isochronous traffic generated by multimedia traffic.

3 System Architecture

In the last section we outlined earlier work on mechanisms for energy efficient multimedia streaming as well as the limitations of client-only policies in conserving energy without losing data packets. In this section, we describe the objectives, the system architecture and the experimental setup. We will highlight our experiences in the next two sections.

3.1 Objectives

Our primary goal was to reduce the energy required by the wireless client to consume a certain multimedia stream (of a given quality). We explore the effectiveness of energy aware traffic shaping in the network infrastructure closer to the mobile client and in the wireless

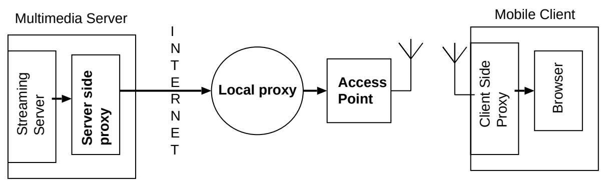


Figure 1: System Architecture

access point itself. The mechanism should also allow energy savings for multiple wireless clients sharing the same wireless access point.

Our experiments were designed to answer the following questions:

- Can we realize energy savings at the network MAC level without an understanding of the application level stream dynamics?
- Can traffic shaping assistance from the network infrastructure allow the client to transition to lower power consuming states more effectively?

3.2 Architecture

The system architecture is illustrated in Figure 1. The system consists of a server side proxy (SSP) or local proxy (LP) and a client-side proxy (CSP). The server side proxy allows the flexibility of traffic shaping at the source, without actually modifying the multimedia servers themselves. The local proxy performs similar functionality to a server side proxy and shares the same LAN network with the wireless client. The server side and local proxies can inform the client-side proxy of the next scheduled data burst. The client-side proxy interacts with the server side or local proxies. It is the responsibility of the client-side proxy to transition the WNIC to a lower power *sleep* state between scheduled data transfers. Since no data transfers are expected during this sleep interval, no data is lost.

3.3 Experiment Setup

The system setup that was used to customize the network transmissions to conserve energy for popular streaming formats is illustrated in Figure 2. The various components of our system are:

- **Multimedia Server**: Our multimedia server (Dell 330) was equipped with a 1.5 GHz Pentium 4 with 512 MB of PC800 RDRAM memory, running Microsoft Windows 2000 Server SP2. The server was running Windows Media Service, Realserver 8.01 and Apple Darwin Server 3.0.1.
- **Wireless Access Point**: For our experiments, we used a dedicated D-Link DWL 1000, Orinoco RG 1000 and Orinoco AP 500 access points. The AP 500 was connected to an external range extender antenna. Throughout our experiments, we had turned off the security encryption feature of the access points.
- **Local proxy**: We used a Dell dual processor (Pentium Xeon 933 MHz) server with 1.5 GB of memory, running FreeBSD 4.4 (STABLE) for the local proxy. The proxy buffered packets from the multimedia servers and transmitted them after the configured delay. The proxy added these delays after transmitting all pending packets.
- **Browser Stations**: We used two 500 MHz Pentium III laptops with 256 MB RAM and running Windows 98 for our browsing stations. Wireless connectivity was provided by 11 Mbps Orinoco PCMCIA WLAN cards. The laptops accessed the streaming formats using Microsoft media, Real and Quicktime players.
- **Monitoring Station**: The packets transmitted from the server to the browser station was passively captured by the monitoring station, which was physically kept close to the browser station and the wireless access point. We used an IBM T21 laptop with 800 MHz Pentium III processor, 256 MB RAM and running Redhat Linux 7.2. Packets were capturing using tcpdump 3.6. We assume that the packets arrive at similar time durations to the tcpdump and the browser applications.

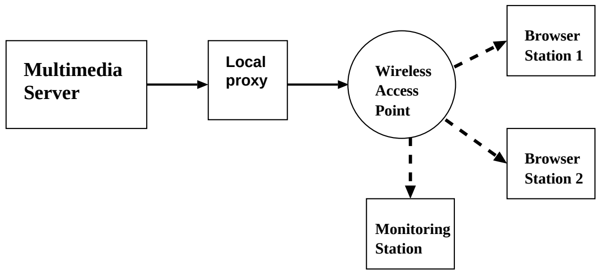


Figure 2: Experiment Setup

The tcpdump packet traces captured by the Monitoring station were fed to our client-side proxy simulator to an-

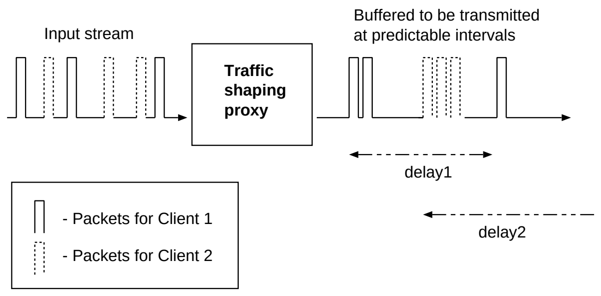


Figure 3: Policies to shape the network traffic to arrive at predictable intervals

alyze the system energy performance without perturbing the Browser stations. We utilized published power parameters [32, 18] for a 2.4 GHz DSSS IEEE 802.11b Orinoco card for our simulations. The model does not simulate lower level energy costs such as unsuccessful attempts to acquire the channel (media contention), or in messages lost due to collision, bit error or loss of wireless connectivity. Further, our simulations do not leverage the battery recovery effect. We assume a linear energy model for wireless NIC power consumption.

3.3.1 Multimedia Stream Collection

For our experiments, we used the Wall (movie) theatrical trailer. We replayed the trailer from a DVD player and captured the stream using the Dazzle Hollywood DV Bridge. We used Adobe Premiere 6.0 to convert the captured DV stream to the various streaming formats. The trailer was 1:59 minutes long. The Wall trailer was digitized to a high quality stream and hence allowed us the flexibility of creating streams of varying formats and fidelities. Hence, we use this stream for the rest of this paper.

3.4 Traffic shaping policies in the network infrastructure

The various states of packet transmission for a traffic shaping network proxy is illustrated in Figure 3. The proxy buffers network packets from different flows (clients) and transmits them at client-specific intervals. The server maintains separate delay intervals per individual client (e.g. delay1 and delay2) and transmits packets to avoid contention on the wireless network. Such traffic shaping allows multiple clients to operate without interfering with each others' sleep intervals.

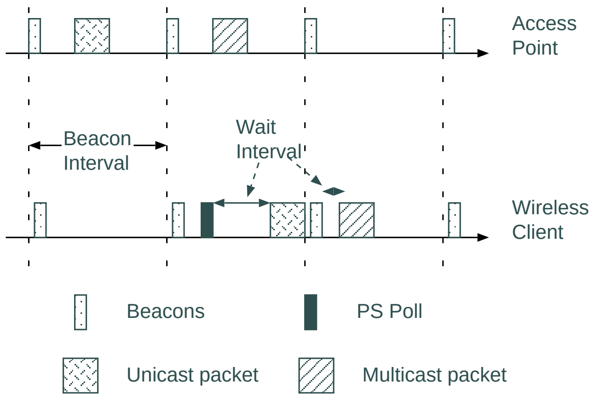


Figure 4: IEEE 802.11 Power Saving Mode (simplified)

3.5 Performance metrics

For our experiments, we use the following performance metrics to measure the efficacy of our approach:

- **Energy consumed:** The goal of these experiments is to reduce the energy consumed by the WNIC.
- **Energy metric:** Depending on the delay introduced by our traffic shaping policies, the multimedia players automatically (and incorrectly; since the effective bandwidth available was still the same) adapted to the delays by lowering the stream fidelity. In order to compare the energy consumption in such a scenario (where the amount of data received can be different), we introduce the notion of energy metric; defined as the amount of energy required to download multimedia data (denoted in Joules/KB). It is preferable to decrease the energy metric. In general, even though low fidelity streams consume less overall energy, the energy metric is high as the WNIC's spend most of the time in wasted *idle* or *sleep* states (instead of actually receiving useful data).

4 Implications of IEEE 802.11 Power Management in Wireless Access Points

First we explore the effectiveness of IEEE 802.11 MAC level power saving mode for conserving the client WNIC energy consumption. In the next section, we investigate a proxy architecture to effect energy savings.

The IEEE 802.11 wireless LAN access standard [23]

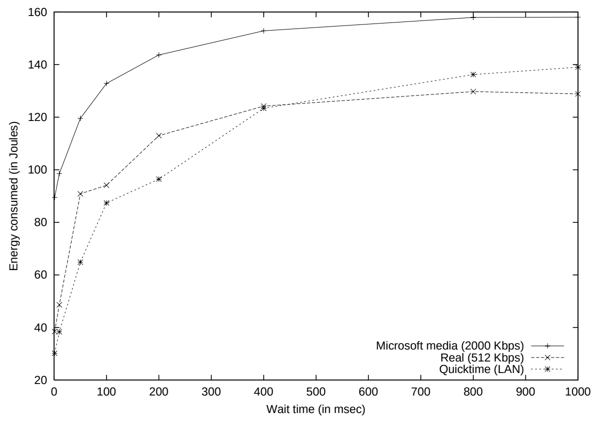


Figure 5: Energy consumed for varying *Wait* times

defines a power saving mode of operation wherein the wireless station and the access point cooperates to conserve energy (illustrated in Figure 4). The wireless station informs the access point of its intention to switch to power saving mode. Once in power saving mode, the wireless station switches the WNIC cards to a lower power *sleep* state; periodically waking up to receive beacons from the access point. The access point buffers any packets for this station and indicates a pending packet using a traffic indication map (TIM). These TIMs are included within beacons that are periodically transmitted from the access point. On receipt of an indication of a waiting packet at the access point, a wireless client sends a PS poll frame to the access point and waits for a response in the active (higher energy consuming) state. The access point responds to the poll by transmitting the pending packet or indication for future transmission. The access point indicates the availability of multiple buffered packets using the *More* data field. For multicast and broadcast packets, the access point transmits an indication for pending packets using a delivery TIM (DTIM) beacon frame; immediately followed by the actual multicast or broadcast packet (without an explicit PS poll from clients). DTIM intervals are usually configurable at the access point and can be any multiple of the beacon interval. The standard does not define the buffer management or aging policies in the access points. Note that the standard does not define a power saving transmit mode for the wireless station itself, it can transmit a packet whenever it wants (regardless of the beacon).

At first glance, it would appear that the 802.11 power saving mode can indeed allow the wireless clients to conserve energy by allowing them to frequently transition to lower power consuming *sleep* state. However, the potential energy saving depends on minimal *Wait* interval (time between the transmission of PS poll message and receipt of the data packet; illustrated in Figure 4). The IEEE 802.11 standard does not specify any time bound for this *Wait* interval. For a general purpose wireless access point, reducing the *Wait* interval has the inadvertent side effect of increasing the priority of data packets for wireless stations operating in the power saving mode. Access point manufacturers typically associate power saving mode as a lower throughput state. They also discourage high rate multicast traffic for the same reason.

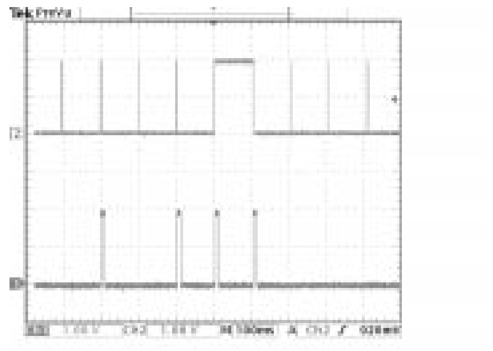
In order to understand the implications of this wait interval, we developed a simple simulator that modeled the various power saving states of a popular WNIC. We modeled a 2.4 GHz Lucent wireless card that consumes 177 mW, 1319 mW and 1675 mW while in *sleep*, *idle* and *read* states, respectively. These energy parameters were published in [32, 18]. We chose a beacon interval of 100 ms (used by Orinoco access points) and varied the average *Wait* times for reasonable values of 0 through 1000 msec. The access points are modeled with infinite buffer space; as the wait times increase, more packets are buffered and are sent back-to-back in succession. A realistic access point would drop these packets once the buffers fill up.

We plot the results for varying the average *Wait* interval for some of the streams measured in our earlier study ([5]) in Figure 5. For comparison, in [5], we noted that a client-side technique consumes 135 (0.15% data loss), 132 (8% data loss) and 47 (23% data loss) Joules for these MS Media, Apple Quicktime and Real streams, respectively. To receive these streams, the WNICs needed to be in active *read* state for 43.63%, 11.30% and 5.11% of the time, which corresponds to a necessary energy consumption of 90.24, 42.94 and 33.57 Joules, respectively. From Figure 5, we note that the potential energy saving is heavily dependent on the average *Wait* intervals. Wait times of zero illustrates the necessary energy consumption values. For average wait times less than 100 msec, even small increase in average *Waits* can drastically affect the energy saving. However, larger wait times do not offer much energy savings.

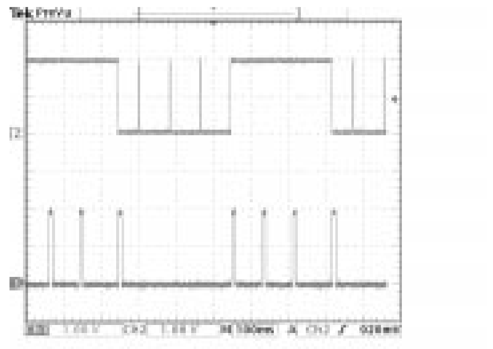
Hence, we tried to measure the actual *Wait* times for typical access points. We carefully disassembled the plastic shielding around a Orinoco Silver PC card and hooked two voltage probes around the two status LEDs. The first LED showed the card power state (transitioning to a high voltage stage while the card is active) while a second LED showed when a data packet was transmitted or received. We connected the PC card to an external Orinoco range extender antenna to reduce the effects of our instrumentation on the proper operation of the wireless card. We used this instrumented PC WNIC card on laptops running MS Win 2000, Win 98, Redhat Linux 7.2 and Compaq iPAQ; configured the PC card to operate in power saving mode and watched media streams using MS Media, Real and Apple Quicktime players for the Windows machines, Real for Linux and PocketTV and MS Media players for the iPAQ. For our study, we used Orinoco RG 1000, Orinoco AP 500 and D-Link DWL 1000 access points. All the access points were set up on a dedicated LAN segment (with no background network traffic) and operating on the same wireless channel.

We noticed that Orinoco RG 1000 and AP 500 access points used a beacon interval of 100 msec (in spite of our attempts at changing this interval; the Linux drivers provides an API to request a different beacon interval). The D-Link DWL 1000 allowed us to choose an interval of either 160 msec or 80 msec; the Windows driver did not allow us to modify this interval and chose 80 msec for TIM. We plot several representative results for the card status during our experiment in Figure 6. Note that the different plots show different parts of the video stream. We show the results for several low bitrate streams. These lower quality streams transmit less date and can be expected to offer considerable energy savings. In each graph, the plot for Channel 2 (top) shows the duration that the card stays in active state. Channel 1 (bottom) shows the duration when an actual packet is either transmitted or received at the card. Ideally, we want the top plot to be active for the least amount of time; appearing similar to the bottom plot.

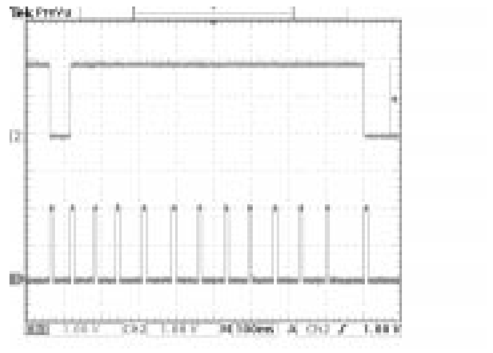
Figure 6(a) plots the results for viewing a MS stream at 56 kbps using the iPAQ device. From Figure 6(a), we note that the power save mode sometimes works optimally. The first two data packets were received with the card in higher power state for the least amount of time. The third packet however triggers the card to stay in higher energy consuming state for a long time, the access point does not transmit the next packet until the next beacon interval (a *Wait* interval of 100 msec). Figure 6(b) plots the results for viewing a Apple Quicktime streaming at 30 kbps from a laptop. For Figure 6(b), we notice similar periods of active waiting. Also in Figure 6(c), we notice longer wait intervals for watching a Real stream at 56 kbps. As noted in [5], Real tends to spend many smaller packets (as compared to Microsoft media). Such packets tend to leave the WNIC at higher energy consuming active state while the access points tries to operate “fairly” for a general audience. In fact, we noticed that watching any stream over 56 kbps tends to completely leave the WNIC in higher energy consuming active state (even though it must be possible to keep the *Wait* times lower, albeit consuming most of the avail-



(a) iPAQ, MS Media stream at 56 kbps, Orinoco AP500 Access point



(b) Windows 98, Apple Quicktime stream at 30 kbps, D-Link DWL 1000 Access point



(c) Windows 2000, Real stream at 56 kbps, Orinoco AP500 Access point

Figure 6: Status of IEEE 802.11b wireless PC card in powersave mode (the top plot (Ch 2) shows the WNIC power state and the bottom plot (Ch 1) shows data transmission/reception intervals. The x-axis shows the time in 100 msec ticks with voltage along the y-axis)

able bandwidth).

We also experimented with the power implications of receiving multimedia streams using multicast packets. The Orinoco drivers allow us to configure the DTIM interval (higher values would reduce the multicast throughput). We observed that even when the DTIM interval was the same as TIM interval, the access points tended to drop the multicast packets. In particular, the D-Link DWL 1000 access points dropped most of the multicast packets. Hence multicast was not a reliable streaming mechanism for our purposes.

In general, we believe that the 802.11b power saving mode has limitations for saving client WNIC energy consumption for the following reasons:

1. **Access point behavior hardware dependent:** The potential energy saving depends on the policy choices at both the access point and the mobile client station. A general access point needs to balance the need for power saving on a single client with fairly sharing the available bandwidth. Even with no other clients to share the bandwidth, the access points tested still tended to keep the clients waiting for extended periods of time.

2. **Does not co-exist for multiple clients:** For multiple clients accessing multimedia streams simultaneously, the TIM specifies all the clients with pending network packets. The standard does not define the order in which client requests are actually serviced. Clients could wait for the whole time needed to transmit packets for all the clients, even though the PS poll requests were all sent at the same time.

   The mobile station can delay the transmission of the initial PS poll message to allow the access point to transmit data packets for other nodes. However, we are not aware of any such protocol defined by the standard to control the client PS poll interval.

3. **Uses fixed TIM interval for all clients:** Even though the standard does not explicitly prevent the access point from dynamically varying the beacon interval to accommodate the observed traffic levels, none of the access points that we tested changed the beacon interval. The choice of the beacon interval is a trade-off between the average packet delay at the access point and the periodicity of client wakeup intervals. As the data rate increases, reducing the TIM interval can reduce the packet delay at the access point. With multiple clients in power saving mode of operation, there is a need for per client TIM intervals.

4. **Power save operation not application aware:** With the $notification(TIM) \rightarrow request(PSpoll) \rightarrow datatransmission$ model of operation, the IEEE 802.11b standard assumes that the data traffic is sporadic and well behaved (few packets spread evenly). On the other hand, multimedia streams tend to be isochronous. Formats such as Real tends to transmit smaller packets at close intervals. Formats such as MS media tends to utilize network fragmentation leading to fragmented packets sent closer to each other. Delaying parts of a fragment can delay the delivery of the entire packet to the multimedia browser.

4.1 Whitecap technology and IEEE 802.11e standard

The upcoming IEEE 802.11e will incorporate the whitecap [9] technology to provide QoS guarantees for multimedia. The technology provides contention free access for multimedia traffic by reserving portions of the transmission spectrum for periodic multimedia traffic. It seems possible for clients to operate in power save mode; transitioning to an active state to receive multimedia streams. The standard is still in the draft stages.

5 Implications of energy aware multimedia service

In the last section, we discussed the limitations of utilizing the 802.11b MAC power management scheme for popular streaming media formats. In this section, we explore the implications of modifying the origin server (through a proxy) to customize the streams so that the clients can frequently transition the WNICs to lower energy consuming states.

In general, any traffic shaping at the origin server will be affected by the loss and delay characteristics of the wide-area Internet. Traditionally, buffering at the client had been used to offset these delays. However, the clients need to know the exact time of arrival for the first packet in order to minimize data loss. The loss of the last packet in a stream (which indicates the arrival of the next packet stream) affects the potential energy saving. Also, multiple clients using the same access point but receiving different streams would experience delays because of competing data reception characteristics. In summary, modifying the origin servers to shape the network traffic

Table 1: Energy consumed and % packets dropped by a client-side history based approach (detailed discussion in [5])

Stream Format	**Stream b/w**	**Energy**	**Client-side adaptation**	
	(in Kbps)	**(in Joules)**	**Energy (in Joules)**	**Bytes dropped (%)**
Microsoft Media	56	157.6	35	1
	128	157.7	52	2
	256	160.2	60	0.5
	768	163.2	70	0.25
	2000	174.8	135	0.15
Real	56	119.2	116	4
	128	119.2	82	11
	256	124.3	120	5
	512	133.2	132	8
Quicktime	56	149.7	41	30
	128	150.1	38	38
	256	149.2	47	23

can enable clients to frequently transition to lower power states has the following drawbacks:

- Loss of the control packet in a stream can adversely affect the energy savings. Throughout this work, we assumed that the wireless network does not suffer from noticeable multimedia data packet loss. In general, if the control packet specifying the client sleep interval from the local proxy is lost, then the client will wait in a higher power consuming *idle* state. Losing the data packets itself would trigger high level mechanisms that adapt the stream to a lower fidelity stream.
- Multiple local clients receiving streams from different servers can lead to conflicting schedules on the network.
- Any packet delay in the wide area can leave a client in a higher power consuming state.

We implemented our policies on a local proxy based architecture. The local proxy buffers the multimedia packets and periodically transmits them to the client (similar in spirit to the IEEE 802.11 beacons). The local proxy could also dynamically inform the client-side of the next packet arrival time using a special control packet. The client-side proxy uses the interval between transmissions to transition the client WNIC to a lower energy consuming *sleep* state. The client-side proxy informs the local proxy server of the access point that it is associated with. The local proxy schedules the transmissions in such a way to avoid conflicts with other clients using the same wireless access point.

First we analyze the energy savings for the various popular streaming formats (MS Media, Real and Apple

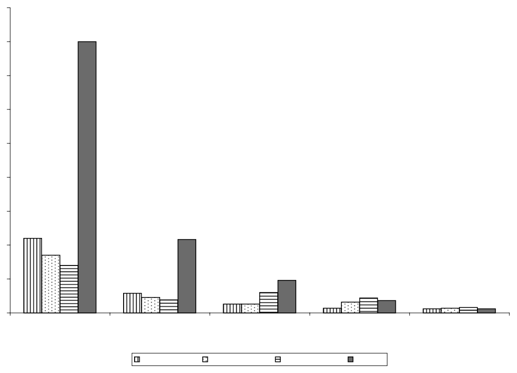


Figure 7: Energy metric for MS media streams

Quicktime) using streams customized for different network bandwidth requirements. We also explore the implications of multiple clients sharing the same wireless access point. For reference, the energy savings and the associated data loss for the various streaming formats using a client based adaptation strategy (that was discussed in [5]) is tabulated in Table 1.

5.1 Single wireless client associated with the wireless access point

First we perform experiments to explore the energy saving possible with a cooperating proxy that enables the clients to transition to lower energy consuming *sleep* state. We explore the implications for popular streaming formats (Microsoft media, Real, Apple Quicktime). We configured the local proxy to transmit buffered packets after a delay of 50, 100 and 200 msec.

In some cases (e.g. high fidelity streams), depending on

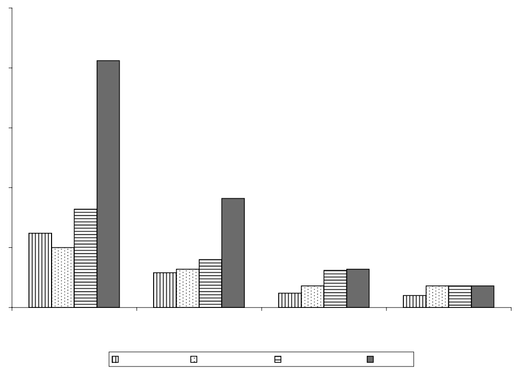


Figure 8: Energy metric for Real streams

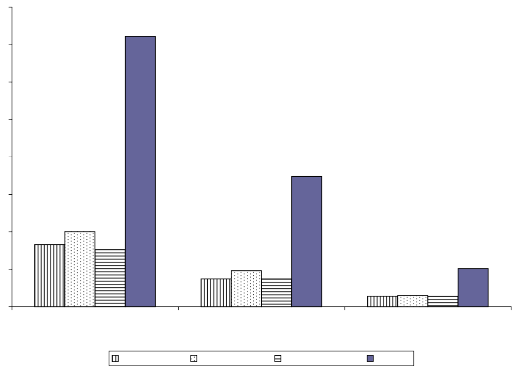


Figure 9: Energy metric for Quicktime streams

the delay introduced, the multimedia players automatically (and incorrectly) adapted to the delays by lowering the stream fidelity. In order to compare the energy consumption in such a scenario, we used the notion of energy metric, defined as the amount of energy required to download multimedia data (denoted in Joules/KB). It is preferable to reduce the energy metric.

5.1.1 Microsoft media streaming format

We plot the energy metric for video streams customized for 56 kbps, 128 kbps, 256 kbps, 768 kbps and 2000 kbps bandwidth streams in Figure 7. From Figure 7, we note that a proxy that transmits packets every 200 msec can offer energy savings for low fidelity stream streams (without any data loss associated with a client-only scheme [5]). For a 56 kbps stream, a server that transmits packets every 200 msec can reduce the energy consumption by as much as 83%. However, for high bandwidth streams (low bandwidth streams are themselves transmitted infrequently) and increased delays, the media player adapts to increasing delays by reducing the stream fidelity. This results in reduced energy consumption while increasing the energy metric. In fact, the energy metric can sometimes be worse than (e.g. 768 kbps stream) receiving the original unmodified streams. Such adaptation can be counteracted by buffering the network packets in the client-side proxy and locally delivering them to the multimedia player at a more regular pace. Note that such additional buffering introduce their own energy requirements for maintaining the local buffers.

Also, we used the *nanosleep()* system call in the local proxy to delay packets for delivery. General purpose operating systems such as FreeBSD and Linux schedule sleeping jobs at 10 msec intervals and hence the actual sleep intervals can be at least 10 msec more than the programmed value. We noticed that this extra interval tends to be more than 10 msec for processes that sleep for longer intervals of time. The client-side proxy actively waits for packets in this extra 10 msec in a higher energy consuming *idle* state, leading to increased energy consumption and higher energy metric. Real time schedulers for Linux can reduce this scheduling interval to 2 msec. We are currently investigating such schedulers to further reduce the energy consumption.

5.1.2 Real streaming format

We repeat the experiments from last section for Real streams transmitted at 56 kbps, 128 kbps, 256 kbps and 512 kbps and plot the corresponding energy metric in Figure 8. We configured the real player to not transmit the stream reception quality feedback to the origin servers and hence the system did not adapt to the packet reception delays. We noted that the server assisted policies offer substantially better energy saving than simple client-side only policies without any associated data loss. For example, a 56 kbps stream with a local proxy that transmits packets every 100 msec only consumed 28 Joules (as compared to 116 Joules and 4% data loss for client-side history based mechanisms). Also, recall that Real typically streams the video streams quicker than the other formats. Introducing high delays (e.g. 200 msec) prolonged the stream transmission, adding extra *sleep* cycles and slightly increasing the energy metric.

5.1.3 Quicktime streaming format

In the last two sections, we explored the potential energy saving in a server that transmits the stream packets in predictable intervals for Microsoft media and Real streams. In this section, we repeat the experiments for

Table 2: Energy consumed and % packets dropped by the client-side history based approach for 2 simultaneous clients in the same wireless access point

Stream Format	Stream b/w	Energy (in Joules)	Bytes dropped (%)
Microsoft Media	56 Kbps	37.00	0.79
	128 Kbps	54.03	0.09
	256 Kbps	57.98	51.77
	768 Kbps	72.74	19.77
	2000 Kbps	135.34	10.17
Real	56 Kbps	39.63	39.10
	128 Kbps	58.23	24.74
	256 Kbps	79.29	21.49
	512 Kbps	96.35	34.42
Quicktime	56 Kbps	27.17	29.71
	128 Kbps	27.96	49.22
	256 Kbps	32.94	41.49

Apple Quicktime streams at 56 kbps, 128 kbps and 256 kbps and plot the results in Figure 9. From Figure 9, we notice the potential energy savings without the associated high data loss rates (illustrated in Table 1). Note that Quicktime transmits the audio and video portions of the stream using separate UDP stream channels. The local proxy needs to be aware of these independent streams in order to schedule the data transmissions.

5.2 Implications of multiple clients using the same wireless access point

In the last section we explored the potential energy savings for a simple server policy that transmits packets at predictable intervals (similar to the IEEE 802.11 MAC level power saving mode). We performed our experiments on a dedicated WLAN environment with a single wireless client. We showed that the potential energy savings over client-side policies. We discussed the effects of operating system scheduling policies that can reduce the potential energy saving. We also identified media player adaptation to increased packet delay and their effects on energy metric.

However, in a typical operating scenario, there could be many mobile clients within a single access point sharing the same physical medium. The local proxy with the knowledge of the physical WLAN limitation can schedule the clients such as to minimize data contention at the network. In this section, we explore the implications of two clients using the same wireless access point, consuming the same video stream (slightly offset in time to avoid the same data from being multicast just once).

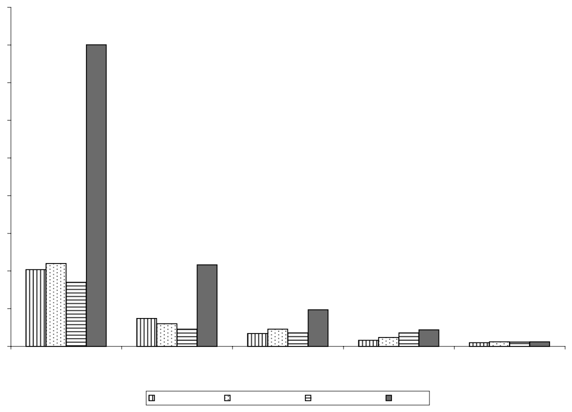


Figure 10: Energy metric for two simultaneous MS media streams

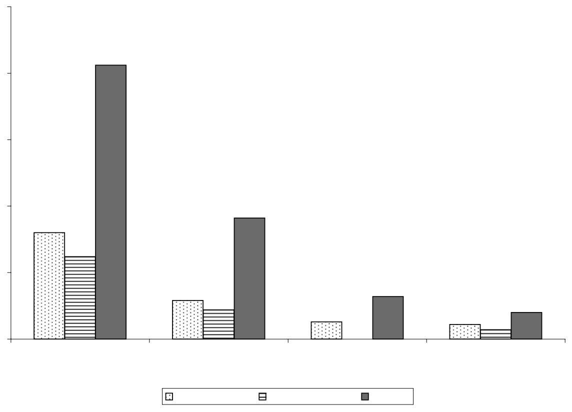


Figure 11: Energy metric for two simultaneous Real streams

First we tabulate the energy saving and the associated data loss for a client-side history based policy (described in [5]) in Table 2. Table 2 shows the inherent limitations of a client-side history based policies in a network with high contention. Both Real and Quicktime streams as well as high bandwidth Microsoft media streams experience higher data loss compared to a single client case shown in Table 1. Low bandwidth Microsoft media streams are transmitted at fairly regular intervals which are not affected much by the increased network contention.

5.2.1 Microsoft media streaming format

We performed experiments with a local proxy servicing two clients accessing the same stream of identical stream quality. We plot the energy metric for the various Microsoft media streams in Figure 10. We notice similar results as in the single client case (Figure 7), showing the potential for a application level contention reduction

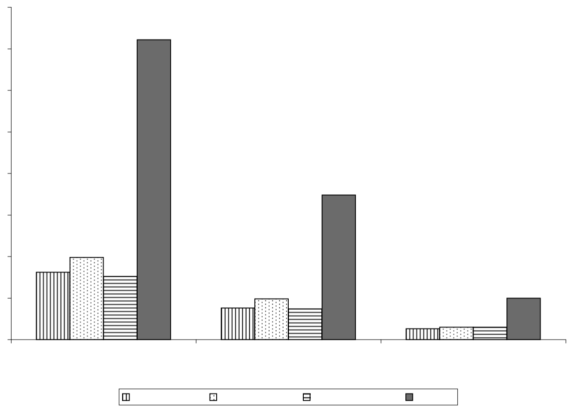


Figure 12: Energy metric for two simultaneous Quicktime streams

mechanism for energy conserving policies. Note that our WLAN only supports an effective throughput of about 4 Mbps. Using two 2 Mbps streams saturates the wireless network; adding additional delays to the steams worsens the contention interval leading to the player adapting to a lower bandwidth stream.

5.2.2 Real streaming format

We repeated the experiments for Real streams and plot the results in Figure 11. For the most part, two clients provide similar savings as a single client case. However, Real players inexplicably experience fatal errors particularly operating with a delay of 50 msec. We are presently investigating why the Real players crash when two players are simultaneously accessing two multimedia streams.

5.2.3 Quicktime streaming format

We continue with our analysis for the various Quicktime streams and plot the results in Figure 12. From Figure 12, we notice similar performance gains for two clients simultaneously accessing multimedia streams. Again, we notice that Quicktime clients adapt to any introduced delays by lowering the stream quality. A caching client-side proxy can offset this client behavior.

In this section, we showed the effectiveness of application aware local proxy in shaping the multimedia traffic. The system can not only allow the clients to better manage their energy, but also schedule the packets to avoid local WLAN network contention. The Quicktime multimedia players adapt to any introduced delays by lowering the stream quality. A caching client-side proxy can help offset such client behavior.

6 Discussion

In this paper, we show the limitations of IEEE 802.11 power saving mode for receiving popular multimedia streams (Microsoft media, Real and Quicktime). We showed that the non-deterministic TIM interval and the associated *Wait* interval can adversely affect the potential energy savings. We showed that the WNICs effectively switch out of the power saving modes for even moderately high bandwidth streams (128 kbps). Also, a single TIM can reduce the energy savings for multiple clients contending for the same TIM beacons by increasing the wait intervals for each stream.

On the other hand, an application-specific server side traffic shaping mechanism can offer good energy saving for all the stream formats without any data loss. We use a simple server enhancement to transmit network packets at predictable intervals. We show that we can reduce the energy metric (Joules/KB) by as much as 83%. We show how our approach can offer similar benefits for two clients sharing the same wireless access point. We note that the operating system scheduling mechanisms induce additional latencies that reduce further potential energy savings. Also, Quicktime players adapt to any network delays by lowering the stream quality.

Our work makes the following contributions towards the design of such application specific energy-aware network traffic shaping mechanisms:

- We show that the amount of energy saving delays introduced depends on the stream requirements; lower fidelity streams are more tolerant to longer delays.
- Operating system scheduling mechanisms can restrict choosing too small values of these delays.
- These mechanisms have to take the stream format into consideration. Formats such as Quicktime that transmit a given media stream using at least two independent channels (one each for audio and video) should be treated properly so as to avoid conflicts within the same application.
- Additional information such as the associated access point can also help avoid network media contention issues.

- Future media players should provide a configurable mechanism for specifying wireless networks such that these energy saving transitions do not trigger network congestion response.

Acknowledgments

We thank Ben Bishop for his help with the oscilloscope. We especially thank our reviewers and our sherpard Mary Baker for all their helpful comments and suggestions. This work was supported in part by a research grant from the Yamacraw initiative.

References

[1] Prathima Agrawal, Jyh-Cheng Chen, Shalinee Kishore, Parameswaran Ramanathan, and Krishna Sivalingam. Battery power sensitive video processing in wireless networks. In *Proceedings IEEE PIMRC98*, Boston, September 1998.

[2] Apple Quicktime. http://www.apple.com/quicktime/.

[3] Mary Baker, Satoshi Asami, Etienne Deprit, John Ousterhout, and Margo Seltzer. Non-volatile Memory for Fast, Reliable File Systems. In *Proceedings of the 5th International Conf. on Architectural Support for Programming Languages and Operating Systems*, pages 10–22, October 1992.

[4] The Bluetooth SIG, http://www.bluetooth.com/. *Specification of the Bluetooth system*, v 1.1 edition, February 2001.

[5] Surendar Chandra. Wireless network interface energy consumption implications of popular streaming formats. In Martin Kienzle and Prashant Shenoy, editors, *Multimedia Computing and Networking (MMCN'02)*, volume 4673, pages 85–99, San Jose, CA, January 2002. SPIE - The International Society of Optical Engineering.

[6] Surendar Chandra, Carla Schlatter Ellis, and Amin Vahdat. Managing the storage and battery resources in an image capture device (digital camera) using dynamic transcoding. In *Proceedings of the Third ACM International Workshop on Wireless and Mobile Multimedia (WoWMoM'00)*, pages 73–82, Boston, August 2000. ACM SIGMOBILE.

[7] Carla Fabiana Chiasserini and Ramesh R. Rao. Pulsed battery discharge in communication devices. In *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking (MOBICOM '99)*, pages 88–95, Seattle, WA, August 1999.

[8] Sukjar Cho. Power Management of iPAQ, February 2001.

[9] Cirrus Logic. Whitecap2 wireless network protocol - hite paper. Technical report, Cirrus Logic, 2001.

[10] Mark D. Corner, Brian D. Noble, and Kimberly M. Wasserman. Fugue: time scales of adaptation in mobile video. In *Proceedings of the SPIE Multimedia Computing and Networking Conf.*, San Jose, CA, January 2001.

[11] Anindya Datta, Aslihan Celik, Jeong Kim, Debra E. VanderMeer, and Vijay Kumar. Adaptive broadcast protocols to support power conservant retrieval by mobile users. In *Proceedings of Data Engineering Conf. (ICDE)*, pages 124–133, September 1997.

[12] Fred Douglis, P. Krishnan, and Brian Bershad. Adaptive Disk Spin-down Policies for Mobile Computers. In *2nd USENIX Symposium on Mobile and Location Independent Computing*, April 1995. Monterey CA.

[13] Carla S. Ellis. The case for higher-level power management. In *Proceedings of the 7th Workshop on Hot Topics in Operating Systems*, Rio Rico, AZ, March 1999.

[14] Keith I. Farkas, Jason Flinn, Godmar Back, Dirk Grunwald, and Jennifer M. Anderson. Quantifying the Energy Consumption of a Pocket Computer and a Java Virtual Machine. In *ACM SIGMETRICS*, pages 252–263, Santa Clara, CA, June 2000.

[15] Laura Marie Feeney and Martin Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proceedings IEEE INFOCOM 2001*, volume 3, pages 1548–1557, Anchorage, Alaska, April 2001.

[16] Jason Flinn and M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *Symposium on Operating Systems Principles (SOSP)*, pages 48–63, December 1999.

[17] Kinshuk Govil, Edwin Chan, and Hal Wasserman. Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU. In *Proceedings of 1st ACM International Conference on Mobile Computing and Networking (MOBICOM95)*, pages 13–25, November 1995.

[18] Paul J. M. Havinga. *Mobile Multimedia Systems*. PhD thesis, Univ. of Twente, February 2000.

[19] David P. Helmbold, Darrell E. Long, and Bruce Sherrod. A Dynamic Disk Spin-Down Technique for Mobile Computing. In *Proceedings of the 2nd ACM International Conf. on Mobile Computing (MOBICOM96)*, pages 130–142, November 1996.

[20] Tomasz Imielinski, Monish Gupta, and Sarma Peyyeti. Energy Efficient Data Filtering and Communications in Mobile Wireless Computing. In *Proceedings of the Usenix Symposium on Location Dependent Computing*, April 1995.

[21] Robin Kravets and P. Krishnan. Power Management Techniques for Mobile Communication. In *Proceedings of the 4th International Conf. on Mobile Computing and Networking (MOBICOM98)*, pages 157–168, October 1998.

[22] Robin Kravets, Karsten Schwan, and Ken Calvert. Power-aware communication for mobile computers. In *International Workshop on Mobile Multimedia Communications(MoMuc'99)*, November 1999.

[23] LAN/MAN Standards Committee of the IEEE Computer Society. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE, 3 Park Avenue, New York, NY 10016, 1999.

[24] LAN/MAN Standards Committee of the IEEE Computer Society. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher-speed Physical Layer Extension in the 2.4 GHz Band*. IEEE, 3 Park Avenue, New York, NY 10016, 1999.

[25] LAN/MAN Standards Committee of the IEEE Computer Society. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher-speed Physical Layer Extension in the 5 GHz Band*. IEEE, 3 Park Avenue, New York, NY 10016, 1999.

[26] Kester Li, Roger Kumpf, Paul Horton, and Thomas Anderson. A Quantitative Analysis of Disk Drive Power Management in Portable Computers. In *USENIX Association Winter Technical Conf. Proceedings*, pages 279–291, 1994.

[27] Jacob Lorch and Alan J. Smith. Software Strategies for Portable Computer Energy Management. *IEEE Personal Communications Magazine*, 5(3):60–73, June 1998.

[28] Microsoft Windows Media Technologies. http://www.microsoft.com/windowsmedia/.

[29] Real Player. http://www.real.com/.

[30] Suresh Singh, Mike Woo, and C. S. Raghavendra. Power-Aware Routing in Mobile Ad Hoc Networks. In *The Fourth Annual ACM/IEEE International Conf. on Mobile Computing and Networking*, pages 181–190, 1998.

[31] Cliff Skolnick. 802.11b community network list. http://www.toaster.net/wireless/community.html, http://www.toaster.net/wireless/aplist.php, 2001.

[32] Mark Stemm, Paul Gauthier, Daishi Harada, and Randy H. Katz. Reducing power consumption of network interfaces in hand-held devices. In *Proceedings of the 3rd International Workshop on Mobile Multimedia Communications (MoMuc-3)*, Princeton, NJ, September 1996.

[33] Amin Vahdat, Alvy Lebeck, and Carla Schlatter Ellis. Every joule is precious: The case for revisiting operating system design for energy efficiency. In *In Proceedings of the 9th ACM SIGOPS European Workshop*, September 2000.

[34] Mark Weiser, Brent Welch, Alan Demers, and Scott Shenker. Scheduling for Reduced CPU Energy. In *USENIX Association, Proceedings of First Symposium on Operating Systems Design and Implementation (OSDI)*, November 1994. Monterey CA.

[35] John Wilkes. Predictive Power Conservation. Technical Report HPL-CSP-92-5, Hewlett-Packard Labs, February 1992.