Choosing Beacon Period for Improved Response Time for Wireless HTTP Clients
Zachary Anderson
Adviser: Srinivasan Seshan
May 1, 2004

1 Introduction

The IEEE 802.11 wireless LAN protocol prescribes the use of a power saving mode in which the network interface card(NIC) of a mobile host is turned off periodically to save energy. The mobile host negotiates this period with the wireless access point, so that the mobile host can have its network interface on to receive beacons. A beacon contains information about the data buffered by the access point for the mobile host while the mobile host's network interface is off. If the mobile host finds that the access point has data buffered for it, it can arrange for the buffered data to be sent.

In practice, the beacon period is typically set to 100ms. For TCP connections, this choice of beacon period can cause observed round trip times to be rounded up to the nearest multiple of 100ms. This rounding effect can hurt performance dramatically [7].

HTTP transactions such as loading the front page of a popular web site typically involve the transmission of several objects under 100KB: images, text, cookies, flash animations, advertisements, etc.. For many websites visited TCP must slow start several connections. During slow start, several RTTs can elapse during which the mobile host is awake but receives no packets. If the 802.11 PSM is used, each RTT will be inflated to 100ms causing a dramatic decline in performance.

Figure 1 shows a cumulative frequency distribution of the sizes of objects retrieved over the course of downloading all top level objects from the web sites listed at the Alexa Top 100 Web Sites page [1]. The average object size was 28KB. Approximately 90 percent of objects were of size fewer than 49KB. Given the small size of the objects, time spent in TCP slow start is a significant part of all transfers.

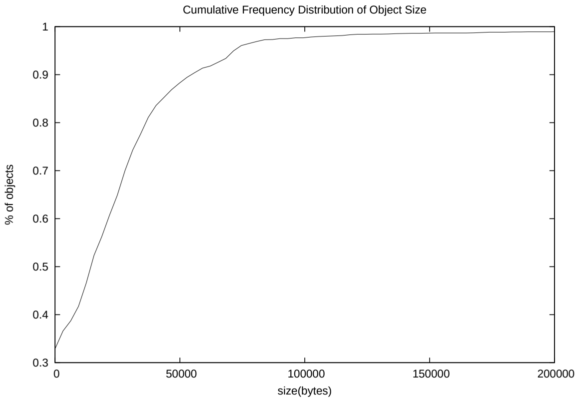

Figure 1: Cumulative frequency distribution of object sizes.

It is important to have some mechanism for power saving, otherwise the network interface can quickly drain a device's battery. Two different techniques have been investigated that try to save power without hurting performance. The Bounded Slowdown algorithm [7] attempts to avoid listening to useless beacons by listening to beacons with decreasing frequency after the mobile host has sent any data. Energy is not saved here during slow start, but this approach has been shown to be effective in simulation with large RTT or with large user thinking time. The second approach involves guessing packet arrival time during slow start [16]. State information about each TCP connection is kept along with an estimate of the RTT. The NIC is put into a low power state after sending the acknowledgment for the last packet in a window, and woken up when the first packet in the next window is predicted to arrive. However, if RTT variance is large, then packet arrival will not be predicted correctly, and packets will be missed, thereby hurting performance.

A third approach, proposed here, is to suppose that the 802.11 PSM allowed beacon periods smaller than 100ms, and to choose the beacon period per TCP connection based on an estimate of the RTT to the remote host. This approach will expose a trade-off between energy efficiency and response time. Our goal is to minimize response time with energy usage similar to that of the 802.11 PSM.

In this paper we investigate the effects of varying beacon period on TCP connections in simulation and on real world TCP traffic through use of an IP traffic shaper, analyze the observed RTT between a large number of remote web servers and a 3Mbps cable connection, and investigate the load on wireless access points in use in the real world. We show that choosing the beacon period based on the RTT results in energy consumption comparable with the 802.11 PSM but at reduced response time. Furthermore, through the analysis of RTT and load on access points, we show that the technique is feasible and generally applicable.

The rest of the paper is organized as follows. Section 2 describes work related to energy efficiency of the wireless network stack. Section 3 describes motivating simulations. Section 4 describes the algorithm in detail. Section 5 discusses some practical considerations in the implementation of the algorithm. Section 6 describes the effects of the algorithm on real Internet traffic. Section 7 summarizes and discusses possibilities for future work.

2 Related Work

A great deal of work has been done on the energy efficiency of all parts of the wireless network stack. The physical layer consists of the Radio Frequency(RF) circuits, modulation and channel coding systems [6]. At this layer research suggests that, due to the lack of comparable advancement in battery technology, efforts be directed at reducing the power used by the NIC [9]. Many energy saving methods at the physical layer are considered in [5].

It was also found in [2] that energy consumption and batter life are heavily dependent on the version of 802.11, the implementation of that version, and the host CPU. It was found that 802.11a and 802.11g consume much less energy for a given workload than 802.11b. The high-rate nature of the 802.11a and g protocols allows them to transmit and receive for shorter periods of time, consuming less energy. Also, different hardware and software designs have different energy properties. The Sony VAIO PCG Z1-AP1 laptop using the Pentium M processor and the Atheros multi-mode MiniPCI 802.11b NIC was found to be twice as energy efficient as Intel's Centrino and the MiniPCI 802.11b NIC. It was also found that power efficient CPUs such as Intel's Pentium M and Transmeta's Crusoe incur smaller penalties for wireless LAN operation than less power efficient CPUs such as Intel's Pentium 4.

There are also a few different Media Access Control(MAC) protocols aside from the 802.11 protocol with different energy properties. In order to be energy efficient a MAC protocol should minimize the number of collisions. This in turn reduces the number of retransmissions. The EC-MAC protocol [13] avoids collisions during reservation and data packet transmission. Another approach in [13] suggests that the base station publish a transmission schedule so that wireless devices do not need to monitor the channel at all times. While not monitoring the channel, wireless devices can drop back into a low power standby mode. The Power Aware Multi-Access(PAMAS) protocol suggests the use of separate channels for Request to Send(RTS)/Clear to Send(CTS) control packets and data packets [12]. While data is being transmitted over the data channel, a "busy signal" is transmitted over the control channel. Devices can enter a low power state if they detect that they are unable to send or receive packets.

At the network layer, energy-aware routing protocols have been developed. Metrics typically used to analyze ad-hoc routing algorithms are shortest-hop, shortest-delay, and locality stability [15]. However, these metrics can result in the overuse of the energy resources of a small set of mobile devices. In [15], energy aware routing algorithms are developed that used among other metrics: energy consumed per packet, and variance in power levels across mobile devices.

Work at the transport layer has concentrated on improving the performance of TCP. TCP was designed to invoke congestion control mechanisms when faced with packet losses. However, invoking congestion control due to a loss caused by the properties of the wireless link can introduce delays and reduce throughput [4]. There have been a few different approaches to this problem. One example is the Snoop Module [3], which uses a split connection protocol. A split connection protocol hides the wireless link from the wired network by having separate TCP connections between the base station and the remote host, and the base station and the mobile device. Furthermore, energy properties of different flavors of TCP are analyzed in [17].

Our work is similar to other work aimed at reducing energy consumption at the transport layer in that we wish to tailor the energy saving algorithm to TCP. It is different from the methods mentioned above in that this goal is reached by entering a low power state during inter-packet arrival times that are on the order of a RTT. Our algorithm has some advantages over a predictive approach found in [16] in that packets may be buffered at the access point if they arrive while the mobile device is asleep. Such packets would be dropped by a predictive approach. Our algorithm has the disadvantage that implementing it on real hardware would require modifying access point firmware. Also, the approach presented here is orthogonal to the Bounded Slowdown approach from [7]. The two approaches could be used together as Bounded Slowdown is primarily concerned with avoiding being awake to listen to beacons when no connections are active. Our approach is primarily concerned only with open connections. The work presented here is independent of energy saving work at other layers of the wireless network stack.

3 Motivation

In this section it is shown through simulation that the default 100ms beacon period for 802.11 PSM is suboptimal. It is possible to reduce response time, but consume an amount of energy comparable to 802.11 PSM if other beacon periods are used.

Simulations were performed using the `ns2` [14] network simulator. The `ns2` simulator implements accurate models of wireless transmission and the 802.11 MAC protocol and so is appropriate for these simulations. The implementation of the 802.11 PSM from [7] was used in these simulations, along with an HTTP traffic generator based on empirical data due to [10].

Figure 2 shows the network topology that was used in simulation. The wireless link is between the nodes labeled `BS` and `WL`, and has bandwidth of 2Mbps and a delay of 1ms. The node `BS` is connected to `R2` by a link with bandwidth 10Mbps and a delay of 1ms. Each `CCi` is connected to `R2` by a 10Mbps link with 2ms delay. `R2` is connected to `R1` by a 100Mbps link with 20ms delay. Each `Wi` is connected to `R1` by a 10Mbps link with 2ms delay. Cross traffic is provided in the form of `HTTP` connections between `CC1` through `CCn` and `W1` through `Wn`. Node `WL` conducts `HTTP` transactions with `Wn+1`.

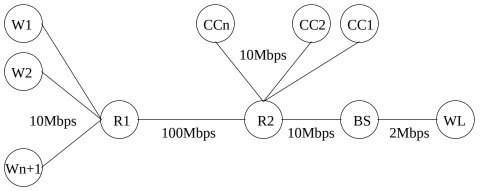

Figure 2: Simulation network topology

Measurements were taken of the connections between `WL` and `Wn+1`. The energy model used was based on the power consumption properties published by Dell about their MiniPCI TrueMobile WirelessLAN 1150 NIC. This NIC consumes 99mW in sleep mode, 759mW in listen mode, 660mW while idle, and 1089mW while transmitting.

The metrics used to evaluate performance were the amount of time used on average to load an entire web page from the remote host, and the amount of energy required on average to do so. These are good metrics because these are the factors that the user of a wireless device is concerned about. Pages should load quickly and a small amount of energy should be used in the process.

Results of the experiments are presented in Figure 3 and Table 1. It can be seen from the graph that both the energy use and time per page load reach a local minimum when the beacon period is set a few milliseconds larger than the round trip link delay. When queuing delay is accounted for, this value is seen to be the RTT observed by the wireless device when no power saving is used. The minimum falls here because RTTs during which there is no traffic during TCP slow start are spent asleep. When the mobile device wakes up again to receive a beacon, packets will just be arriving from the remote host, or will have been buffered for a short period of time to then be delivered by the access point.

In simulation, using a 100ms beacon period incurs a penalty of 430ms per page load, but saves 120mJ per page load. Using a beacon period near the RTT(52ms) incurs a penalty of 120ms per page load, but saves 110mJ per page load.

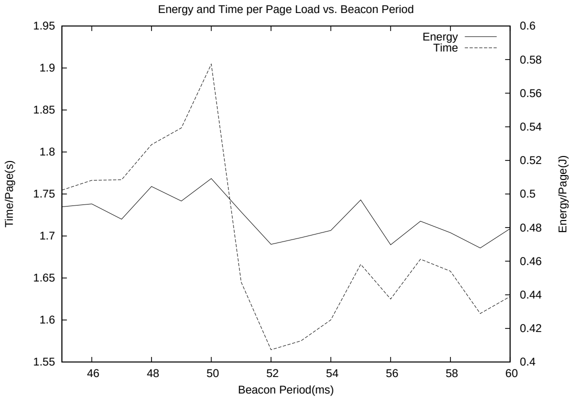

Figure 3: Energy per page load, and time per page load vs. beacon period

We would like to exploit this observation by allowing each mobile host to choose its own beacon period. This choice should be made independently of other mobile hosts populating the same access point.

Method	Average Time/Page Load(s)	Average Energy/Page Load(mJ)
No PSM	1.44	860
52ms Beacon Period	1.56	470
100ms Beacon Period	1.87	460

Table 1: Energy and time statistics

4 Algorithm

In this section the algorithm is described in detail. The algorithm presented here is nearly identical with the 802.11 PSM with the exception that the wireless device may choose the beacon period to be any number of milliseconds. Exploiting the results of the above simulations, we would like to choose a beacon period per connection based on the RTT of the connection.

RTTs can calculated by the wireless device for packets that have not been buffered at the access point. The beacon period is based on a moving average of RTTs so that the mobile device may adapt to changing network conditions. The RTT estimate is maintained in the same way that many implementations of TCP maintain it; as an exponential weighted moving average: $\texttt{ERTT} = \frac{7}{8}\texttt{ERTT} + \frac{1}{8}\texttt{SRTT}$, where `ERTT` is the RTT estimate, and `SRTT` is an RTT sample. The mobile device may periodically inform the access point of the RTT estimate so that a good beacon period can be used for each connection.

Furthermore the mobile device may keep a cache of RTT estimates so as to avoid calculating an estimate on the fly. In the absence of an entry in the cache for a particular remote host, the mobile device may leave its NIC in the higher power state so that an RTT estimate can be calculated–based on the time between sending a SYN packet and receiving and ACK packet–before informing the access point of the desired beacon period.

The access point will need to maintain state per connection, and will additionally need to send beacons for each connection. The increased traffic caused by additional beacons is discussed in the next section.

Despite the fact that this is a conceptually simple modification to the standard 802.11 PSM, there are a couple of complications. Questions still remain about the variance of RTT, and the effects of increased communication and computation load on access points. These concerns will be addressed in the following section.

5 Practical Considerations

This section will address the concerns mentioned in the previous section. First there is the question of whether or not it is feasible to estimate the RTT based on the values contained in the TCP time stamp option. Second is the question of whether or not the increased load on access points will be a problem

5.1 RTT measurements

The algorithm described in the previous section relies on maintaining an accurate RTT estimate. This estimate is kept by calculating an exponential weighted moving average as described above. Therefore, we would like to verify that even with a low bandwidth cable connection, variance in RTT is manageable.

Round trip times were measured at the client side by subtracting the echoed TCP time stamp from the current time. Measurements were carried out between a Comcast digital cable connection in Pittsburgh, PA with bandwidth 3Mbps, and 100 popular and academic websites. RTTs were measured over the course of several `HTTP` connections while loading the front page of the websites and all top level objects. A typical cumulative frequency distribution is shown in figure 4. The average RTT was 57ms and the standard deviation was 37ms.

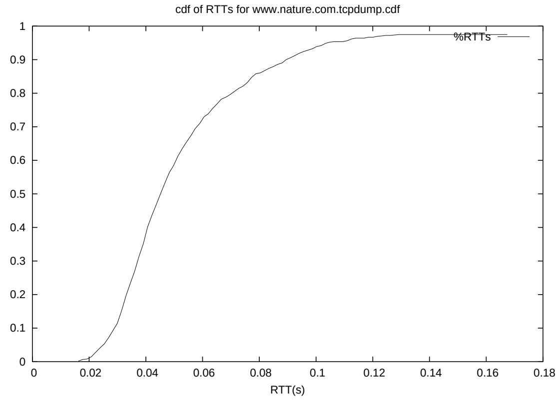

Figure 4: Cumulative frequency distribution of RTTs to www.nature.com.

Even with a connection where larger RTT variance is expected, 90 percent of RTTs are smaller than 100ms such that standard 802.11 PSM will be suboptimal. Also variance is small enough that it is reasonable to expect that we can choose a beacon period that gives performance gains over standard 802.11 PSM. This expectation will be verified in section 6.

5.2 Access Point Population

Additionally, the algorithm described in the previous section relies on the access point sending beacons for each outstanding connection. In order to determine the feasibility of this requirement we present an analysis of access population in a real world deployment.

Population data was gathered from access points in the Graduate School of Industrial Administration building on the Carnegie Mellon campus. The data was gathered over the course of a work week from 18 access points. Figure 5 shows a cumulative frequency distribution of access point population. Roughly half of the time the access point was populated at all the population was only one. On average, 90 percent of the time an access point was populated at all, the population was fewer than 10. It should be noted that population provides an upper bound on load on the access point; the actual number of concurrent connections will be less than or equal to the population.

The increased traffic caused by additional beacons is acceptable because access point population is typically small, and the per connection beacons may consist of only a few bytes. Also, in the rare case that access point population is large, the access point could be configured to revert back to using the same beacon period for each client. This beacon period need not be the standard 100ms, but could be a value based on the beacon periods being requested by the clients. Also, access points could enforce a minimum beacon period in order to keep the wireless channel from being flooded with beacons.

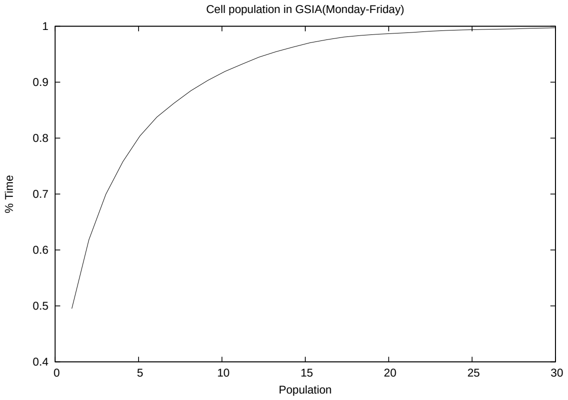

Figure 5: Cumulative frequency distribution of average access point populations.

6 Evaluation

This section describes experiments with real Internet traffic and presents the results of the experiments. Laptops and other mobile devices are frequently used in the wireless setting for web browsing. Also, since existing energy saving techniques perform poorly across many short transfers, attention is restricted to HTTP traffic. A few different techniques were used in experiments with real Internet traffic:

- 802.11 PSM - using the standard 100ms beacon period.
- 802.11 PSM - using a beacon period set to the RTT estimate.
- No power saving - requiring the NIC to be awake at all times.
- Perfect Guessing - assuming that the NIC is always asleep unless sending or receiving packets.

The setup for experiments with real Internet traffic used a 1.7 GHz Pentium laptop running Linux 2.4-18 modified with the KURT real-time patch. The KURT [8] real-time patch was needed so that millisecond resolution timers could be used. The access point was simulated using a modified version of the rshaper IP traffic shaper kernel module [11]. The laptop was connected to the Internet by 3Mbps Comcast digital cable connection.

Also, the kernel is modified such that the value placed in the TCP time stamp option is at millisecond resolution based on the processor's cycle counter–as opposed to the usual jiffy resolution–so that RTTs can be calculated more precisely. The TCP receiver can always calculate the RTT on receiving a packet by subtracting the echoed time stamp from the current time for the first packet received echoing a particular time stamp.

HTTP traffic was generated by accessing the front page and related objects of the 100 most popular websites as listed by the Alexa Top Sites web page. About 3600 objects comprising 7.2MB of data were downloaded over a period of 100 minutes.

In Figure 6, several different beacon periods between 0 and 100 have been used in accessing `superman.web.cs.cmu.edu` with the cable connection. The average RTT to this website was 45ms with standard deviation 20ms. This experiment demonstrates that when RTT variance is high, as it is on a cable connection, a beacon period can be chosen near the actual RTT to achieve performance gains over 802.11 PSM.

In Figure 7, 25 beacon periods evenly distributed between 0 and 100ms were used in accessing the Alexa sites. The figure shows a cumulative frequency distribution of the beacon period that

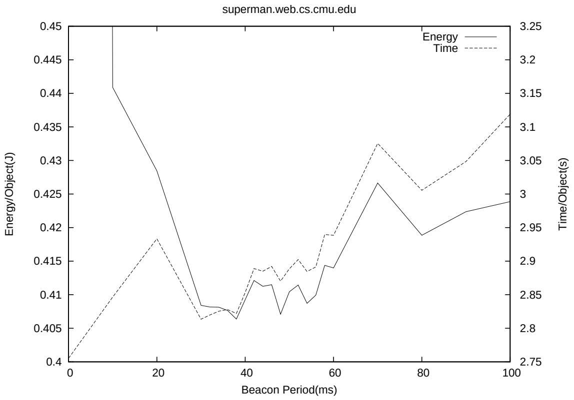

Figure 6: Accessing `superman.web.cs.cmu.edu` with various beacon periods.

minimized the product of average energy used per object and average time spent per object. This demonstrates that when run over real Internet traffic, the algorithm may not simply choose a beacon period around 50ms. The average beacon period chosen in this way was 39ms with a standard deviation of 21ms.

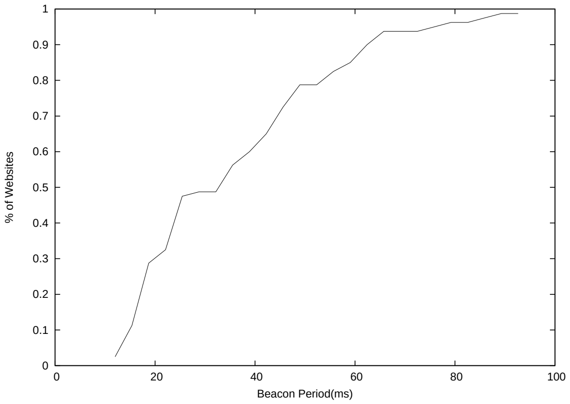

Figure 7: Cumulative frequency distribution of beacon periods minimizing energy per object-time per object product over Alexa top 100 web sites.

Table 2 shows results of different power saving methods used on `superman.web.cs.cmu.edu`. On average, using a 100ms beacon period incurs a penalty of 370ms per object while using a beacon period near the RTT incurs only a penalty of 120ms per object. Also, on average, using a 100ms beacon period saves 1510mJ per object while using a beacon period near the RTT saves 1520mJ per object. Using a 100ms beacon period requires more energy because more time is spent asleep.

Method	Average Time/Page Load(s)	Average Energy/Page Load(mJ)
No PSM	2.75	1930
Perfect Guessing	2.75	390
48ms Beacon Period	2.87	410
100ms Beacon Period	3.12	420

Table 2: Result of techniques on accessing `superman.web.cs.cmu.edu`

Table 3 shows the results of the experiments as the percentage difference between the average time and energy spent per object when the beacon period is set to the estimated RTT and when the beacon period is set to 100ms, when no power saving is used, and with a perfect guessing scheme.

Method	% Time Difference	% Energy Difference
No PSM	0% ± 4%	79% ± 4%
Perfect Guessing	0% ± 4%	0% ± 10%
100ms Beacon Period	14% ± 13%	9% ± 10%

Table 3: Result of different techniques on Alexa Top 100 Web Sites relative to our algorithm

The result of the experiment indicates that both methods using beacon periods save about the same amount of energy, but that the method using a beacon period based on RTT has roughly the same response time as with no power saving or with perfect guessing. Over the course of the experiment using the standard 100ms beacon period incurred on average a 310ms penalty per object, whereas using a beacon period based on the RTT was on par with guessing perfectly and not using power saving.

7 Summary and Future Work

First it was shown through simulation that the standard 100ms beacon period is not always optimal for web browsing. An algorithm was described through which a mobile device may choose a beacon period that is based on an RTT estimate. Then, it was shown that RTT variance is small enough, even with an erratic 3Mbps cable connection, that an effective RTT estimate can be made. Next, it was argued that access points can handle the load incurred by allowing mobile devices to choose their own beacon periods through an analysis of access point population in an actual deployment. Finally the algorithm was used in a modified traffic shaper to access the 100 most popular websites. Results of this experiment indicate that choosing the beacon period to be the RTT estimate saves as much energy as the standard 802.11 PSM, but at much greater performance.

Future work on this topic will include a more detailed study of access point load including an analysis of the number of connections taking place concurrently. Also, RTT measurements and experiments with real Internet traffic will be repeated with a higher bandwidth connection. Furthermore the motivating simulations will be repeated with wireless cross traffic so that the point at which it becomes more efficient to use the same beacon period for each wireless device can be found. Additionally, use of Bounded Slowdown along with our algorithm is expected to provide further energy savings.

References

[1] Alexa top sites. http://www.alexa.com, 2004.

[2] Power consumption and energy efficiency comparisons of wlan products. White paper, Atheros, 2003.

[3] H. Balakrishnan, S. Seshan, and R. Katz. Improving reliable transport and handoff performance in cellular wireless networks. *Wireless Networks*, 1(4), 1995.

[4] R. Caceres and L. Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE Journal on Selected Areas in Communications*, 13(5):850–857, 1995.

[5] A. Chandrakasan and R.W. Brodersen. *Low Power CMOS Design.* Kluwer Academic Press, Norwell, MA, 1995.

[6] Christine E. Jones, Krishna M. Sivalingam, Prathima Agarwal, and Jyh-Cheng Chen. A survey of energy efficient network protocols for wireless networks. *Wireless Networking*, 7(4):343–358, 2001.

[7] Ronny Krashinsky and Hari Balakrishnan. Minimizing energy for wireless web access with bounded slowdown. pages 119–130. ACM MOBICOM '02, 2002.

[8] Kansas university real-time kernel. http://www.ittc.ku.edu/kurt, 2003.

[9] P. Lettieri and M.B. Srivastava. Advances in wireless terminals. *IEEE Personal Communications*, 6(1), 1999.

[10] B.A. Mah. An empirical model of http network traffic. IEEE INFOCOM, 1997.

[11] Alessandro Rubini. rshaper. http://freshmeat.net/projects/rshaper, 1999.

[12] S. Singh and C.S. Raghavendra. Pamas: Power aware multi-access protocol with signalling for ad hoc networks. *Computer Communication Review*, 28(3), 1998.

[13] K.M. Sivalingam, J.-C. Chen, P. Agarwal, and M. Srivastava. Design and analysis of low-power access protocols for wireless and moble atm networks. *ACM/Baltzer Wireless Networks*, 6(1):73–87, 2000.

[14] The VINT Project. *The ns Manual*, 2001.

[15] M. Woo, S. Singh, and C.S. Raghavendra. Power aware routing in mobile ad-hoc networks. ACM MOBICOM '98, 1998.

[16] Haijin Yan, Rupa Krishnan, Scott A. Watterson, and David J. Lowenthanl. Client-centered energy savings for concurrent http connections. Technical report, UGA-CS, 2003.

[17] M. Zorzi and R.R. Rao. Energy efficiency of tcp in a local wireless environment. *ACM/Balter Mobile Networks and Applications*, 2000.