Towards Energy-Awareness in Managing Wireless LAN Applications

Markus Tauber, Saleem N. Bhatti, Yi Yu
School of Computer Science, University of St Andrews, UK
{markus,saleem,yi}@cs.st-andrews.ac.uk

***Abstract*—We have investigated the scope for enabling WLAN applications to manage the trade-off between performance and energy usage. We have conducted measurements of energy usage and performance in our 802.11n WLAN testbed, which operates in the 5 GHz ISM band. We have defined an *effective energy usage envelope* with respect to application-level packet transmission, and we demonstrate how performance as well as the effective energy usage envelope is effected by various configurations of IEEE 802.11n, including transmission power levels and channel width. Our findings show that the packet size and packet rate of the application flow have the greatest impact on application-level energy usage, compared to transmission power and channel width. As well as testing across a range of packet sizes and packet rates, we emulate a Skype flow, a YouTube flow and file transfers (HTTP over Internet and local server) to place our results in context. Based on our measurements we discuss approaches and potential improvements of management in effective energy usage for the tested applications.**

I. INTRODUCTION

The convenience of use of WLAN leads to its use in many devices even those that are not mobile, e.g. new Internet-enabled television sets. At the same time, there exists a very large existing deployment of WLAN systems, spanning a range of evolutions of the IEEE 802.11 standard. Various IEEE 802.11 standards are widely implemented in many consumer devices. This includes devices such as hand-held games consoles, smartphones and televisions, as well as the more 'traditional' uses in laptops/netbooks and desktops.

Meanwhile, there is world-wide concern for the increasing carbon footprint of ICT systems, with current carbon emissions similar to the aviation industry, and emissions set to triple between 2002 to 2012 [1]. So, widespread and growing use of IEEE 802.11 suggests that it is prudent to consider the energy usage of systems incorporating the various 802.11 standards.

A. Motivation

Several power saving mechanisms which aim to enhance energy efficiency have been proposed and are partially implemented: e.g. 802.11 Power Save Mode (PSM) [2], Unscheduled Automatic Power Save Delivery (U-APSD) [3], WMM Power Save (WMM-PS) [4], Dynamic MIMO Power Save [5], and Wake-on-Wireless [6]. However, such features are not widely implemented by all vendors, or are not widely deployed, or are not easily accessible for use by non-expert users. Also, there is a large base of *legacy* (i.e. old) equipment which cannot support such features due to hardware constraints.

Additionally, in situations in which modern WLAN hardware is used to provide key infrastructure (e.g. [7]), providers may be reluctant to jeopardise performance by enabling new energy-saving features [8], [9]. For instance, the generic IEEE 802.11 Power Save Mode (PSM), the 802.11n specific Spatial Multiplexing Power Save (SMPS) and the Power Save Multi-Poll (PSMP) mode implement algorithms which either put the (redundant) WLAN interface into sleep mode or increase buffers. Other approaches to use the WLAN resources more efficiently include packet aggregation. [8], [9].

Another factor impacting the use of power-save modes with sleep mechanisms is the growing popularity of cloud-based services and the use of 'push' service paradigms, giving little opportunity for the user's system to enter sleep mode.

Even as WLAN equipment is upgraded, legacy kit will remain, and the energy efficiencies that might be gained by considering only the NIC, even when expertly configured, are small compared to considering the system as a whole, e.g. transmission power is 1mW-50mW and NIC power usage is low, so NIC-only savings are also low. *Thus, our motivation is to determine the scope for application-level (self-)management of energy usage on a system-side basis.* Our eventual aim is to identify management mechanisms that could be applied to existing (legacy) deployments of 802.11, as well as to new deployments, and is complementary to and independent of, hardware power-saving schemes, such as those listed above.

B. Research focus

We take the position that it is possible to improve the *energy usage* of applications by actions taken at the application level, even if 802.11 power-saving enhancements are not available. So, one of our objectives is to understand the performance and energy usage *dynamics* of existing, commonly-used IEEE 802.11 equipment, with the overall goal that application-level self-adaptation could provide energy efficiencies in 802.11 systems, for example by adjusting packet flow construction and packet transmission [10]. As IEEE 802.11 is so widely used, by applying functionality retrospectively to existing systems, e.g. through software updates and/or patches, we believe that considerable energy efficiencies might be achieved: even small savings for individual devices could have a large impact if applied universally.

Our previous work defines an application-level *effective energy usage envelope* showing the energy usage of IEEE 802.11a and IEEE802.11n at 5 GHz [11]. In this study, we

978-1-4673-0269-2/12/$31.00

focus on IEEE 802.11n at 5 GHz, to identify the *dynamics* of the performance and the effective energy usage envelope in response to changes in transmission power and channel width. Note that we use *off-the-shelf hardware* with default configuration and consider system-wide usage for a node, and not only the NIC within that node. We adopt an empirical approach, taking measurements of power usage for deducing effective energy usage, and measuring system wide performance in our testbed comprising off-the-shelf consumer equipment. While we are concerned primarily with energy usage, analyses only make sense when placed in the context of system performance under different system workloads.

C. Contributions

The contributions of this paper are:

- An empirical study of the energy usage and performance trade-off for a typical WLAN configuration of 802.11n.
- We identify the scope of adaptation that is possible for an application. This acts to define boundaries for what is achievable through management actions or interventions. This is captured in an *effective energy usage envelope*, which indicates the upper and lower bounds of what is possible.
- An analyses of how a (self-)management system for different classes of applications (or how different end-system platforms) might make use of such information in order to provide appropriate application-level adaptation decisions. These could be policy-driven or might be realised as an autonomic system.

We observe that performance and effective energy usage vary greatly between WLAN configurations and traffic profiles, i.e. for different classes of applications. We find that the kind of (self-)management adaptation that is possible may need to be per-user, per-application and context specific. So, MAC layer solutions, such as packet aggregation, that are applied unilaterally might not necessarily provide the best trade-off between energy usage and performance.

D. Structure of this paper

In Section II, we explain our methodology, describe our testbed, our observables and define our energy usage metric. In Section III, we present our observations and discuss what they show. A discussion on how our findings can be exploited to make applications energy-aware and enable self-management is provided in Section IV. We present a summary of related work in Section V, and conclude in Section VI.

II. APPROACH, TESTBED AND METRICS

Our eventual aim is to allow applications to be self-managing. We design our experiments to investigate the scope for self-adaptation of application-level flow characteristics. To do that we adapt the testbed and methodology from our previous work, further details of which can be found in [11]. We consider that most deployed systems are used in 'out-of-the-box' configurations, without specific tuning for energy efficiency. Many WLAN NIC drivers permit various controls of the hardware features, but these might not be easily accessible or comprehensible for modification by most users. A practical constraint we have used is that of a 5 GHz-only testbed. This was because in our local environment, we have exclusive usage of 5 GHz and so the probability for biasses of our measurements by interference was small.

A. Testbed

We have experimentally evaluated energy usage and performance in our 5 GHz testbed. We generated packet flows of offered loads with various bit-rates and packets sizes, and measured power usage during the packet transmission. Our testbed (Figure 1) consisted of a single client host, a host running a wireless access-point (AP) and experimental control units (only one shown in Figure 1) for monitoring the WLAN environment, providing storage for measurement data, *ntp*[1] services and system configuration. The WLAN hosts were setup in a teaching lab in the University of St Andrews with a distance of ∼24±0.5 m between the 2 dBi antennas.

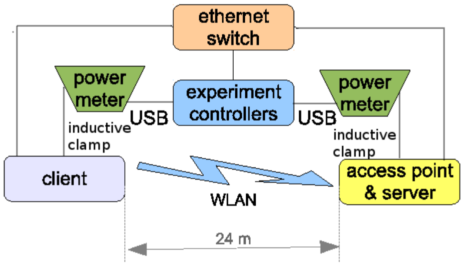

Fig. 1. Schematic of test-bed showing physical connectivity. We used 5 GHz only, and the testbed was configured separately for 802.11n (20 MHz and 40 MHz channels, as well as transmission (TX) powers of 0 dBm and 17 dBm) experiments. The experiment controller uses Ethernet for control messages and shared file-system access. Power meter readings are logged by experiment controllers. The separation between the 2 dBi antennas of the client and access point/server is 24±0.5 m. Data packets generated by *iperf* were transferred across the WLAN link.

We wished to test 802.11n at 0 dBm (1 mW, minimum RF power), 17 dBm (50 mW, typical maximum indoor RF power), and with both a 20 MHz (default) channel and a 40 MHz channel, a total of 4 different combinations. This means that all our experimental workloads in Table I are executed 4 times, once with each of these combinations. Our WLAN card uses the popular Atheros[2] chipset, now bought by Qualcomm so likely to be used even more widely[3]. All machines used Ubuntu 10.04 a minimal server distribution (no desktop service daemons or GUI overhead), with the default kernel 2.6.32-24-generic-pae, and updated WLAN modules (compat-wireless-2011-05-02), which will soon be part of the standard distribution.

---

[1] http://www.ntp.org/

[2] http://www.atheros.com/

[3] http://www.qualcomm.com/news/releases/2011/01/05/qualcomm-acquire-atheros-leader-connectivity-networking-solutions

B. Experiments

Packet generation and performance measurement for UDP traffic was conducted using *iperf*[4] for which the AP was used as the server. A wrapper script at the client executed *iperf* and extracted throughput and loss for individual UDP flows using *iperf server reports*. The specific packet sizes and bit rates of the UDP workload are given in Table I. Motivation for using UDP is its popularity for Voice and Video over IP (VoIP and ViIP) applications and because it allows better control of application specific offered load bit rates compared to TCP, which is modulated by its congestion control behaviour.

TABLE I
GENERIC UDP WORKLOAD.

Packet size	64; 1460 bytes
Bit rate of the offered load	32; 50; 100; 256; 512 Kbps 1; 5; 10; 15; 20; 25; 30; 35; 40; 45; 50; 60; 70; 80; 90 Mbps

Combining packet size and bit-rate gives 40 configurations; 5 measurements for each gives 200 flows; 4 power/channel combinations gives 800 flows; each flow had a duration of 4 minutes, giving over 53 hours of measurements.

We have used emulated flows for Skype, YouTube and HTTP (Internet and Intranet), as summarised in Table II. Traffic emulating a Skype (VoIP) flow was based on previous studies [12], [13], as was traffic emulating a YouTube (ViIP) flow [14], [15]. We do not incur the audio/video codec overhead in our experiments. We have deduced HTTP-specific downstream traffic profiles from preliminary experiments using *wget*[5] to generate HTTP flows from a local server and from http://mirror.ox.ac.uk/ for downloading of an Ubuntu ISO CD image file. For each of the above application specific traffic profiles we have emulated 5 sequential UDP flows with *iperf*. We have used a flow duration of 4 minutes based on previous studies of VoIP and ViIP traffic stated above, but used this duration for *all* traffic workloads for comparability.

TABLE II
APPLICATION UDP WORKLOAD EMULATION.

Skype	300 byte packets, 65 Kbps
YouTube	1431 byte packets, 639 Kbps
HTTP (Internet)	1420 byte packets, 11 Mbps
HTTP (Intranet)	1460 byte packets, 90 Mbps

Combining packet size and bit rate gives 20 configurations; with 4 power/channel combinations gives 80 flows; a flow duration of 4 minutes gives a total of 320 minutes of measurements. Workload based on [12]–[15], as well as on preliminary measurements.

C. Observed variables and metrics

We have measured the observables as described below:

- *Performance:* throughput and loss, as recorded by *iperf*'s server reports on the client for each UDP flow.
- *Power:* every 30 seconds we have recorded the current power consumption in *Watts* at the AP and client.
- WLAN spectrum: the signal strength (showing channel utilisation) as recorded every 30 seconds via the USB-connected spectrum analyser *WiSpy*[6] at one of the experiment controllers. This was for initial calibrations; for further on demand analysis; and also to confirm that during the measurements, only our test-bed was operating at 5 GHz, i.e. to spot possible interference from other sources.

The monitoring intervals for all of the above observables were deduced from preliminary experiments. Motivation for doing this was to avoid excessive disk usage, but having sufficient monitoring sample sets for determining significant differences between experiments.

We measured power consumption on the client and the AP at 30 second intervals. For power measurements, we used a CC128 power meter[7].

For assessing energy usage, we define *effective energy usage (E_A)* as follows:

$$E_A = \frac{\text{mean power used during transmission of flow}}{\text{mean throughput of flow}}$$

E_A has units Joules/Mega-bit (J/Mb):

$$\frac{\text{power in Watts}}{\text{throughput in Mbps}} = \frac{J/s}{Mb/s} = J/Mb$$

and the lower the value of E_A, the better in terms of energy usage. To generate values for E_A, for each individual flow, we use the following measurements:

$$E_A = \frac{P_A - P_I}{T_A} \tag{1}$$

P_A	Mean power consumption measured during the transmission of flow [Watts].
P_I	Mean power consumption measured for an idling node [Watts] (was measured to be 48 Watts).
T_A	Mean throughput measured (using *iperf*) during flow transmission [Mbps].

III. RESULTS AND DISCUSSION

We find that a higher application-specific bit rate results in a lower effective energy usage, i.e low values of E_A. Conversely, low application-specific bit rates result in higher effective energy usage, i.e. high values of E_A. Of course, lower values of E_A are better.

For a given bit rate of offered load, increasing packet size reduces the effective energy usage. This is shown in Figure 2, which plots the mean value of E_A over all experiments (to show trends in effective energy usage), including the operational points of the emulated applications. This is due to amortisation of the transmission overhead and system-wide energy usage across a greater number of transmitted bytes. That is, we have derived E_A based on power measurements for the system has a whole and then evaluated the effective energy usage across the transmitted packets. This is in contrast to other studies that have considered only the NIC, for example. By considering the system as a whole, we are also in a

---

[4] https://sourceforge.net/projects/iperf/

[5] http://www.gnu.org/software/wget/

[6] http://www.metageek.net/products/wi-spy/

[7] http://www.currentcost.com/product-cc128.html

good position to then suggest system-wide (self-)management actions that can be applied at the application level and should then yield reductions in effective energy usage across the system as a whole.

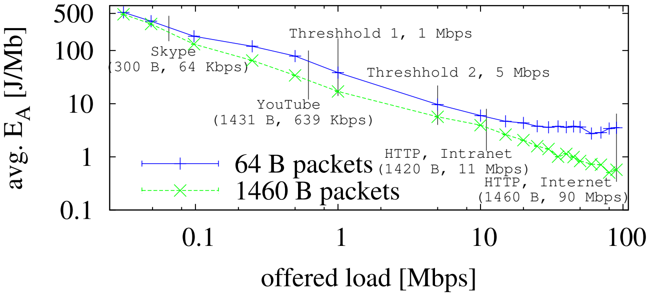

Fig. 2. Mean values of E_A for small and large packets, with operational points of emulated application flows. *Threshold 1* – up to which little/no loss occurs; *Threshold 2* – up to which large and small packets result in the approximately the same throughput.

We have identified a ∼1 Mbps threshold (Threshold 1 in Figure 2) for bit rates of offered load up to which no significant loss nor any difference with respect to throughput can be identified due to different packet sizes. We also observe that up to ∼5 Mbps, there is still little difference in throughput, but there are bursts of loss, up to ∼20 % (Threshold 2 in Figure 2). The figure also shows that E_A ranges from ∼500 J/Mb, for low data rates ($<$ 1 Mbps) with both large and small packets, to ∼3 J/Mb, for small packets with high data rates, and ∼0.5 J/Mb for large packets with high data rates (90 Mbps).

We show in Table III how the highest mean throughput relates to loss and E_A in all of our experiments. Table III shows that high throughput correlates to low effective energy usage E_A (maximum $E_A \sim 500$ J/Mb), but also results in high loss in each of our experiments with high TX power (17 dBm).

TABLE III
HIGHEST MEAN THROUGHPUT, WITH LOSS AND E_A MEASUREMENTS FOR 64B AND 1460B PACKETS, IN ALL EXPERIMENTS, ARRANGED BY CHANNEL WIDTH (CW) AND TX POWER (TX). POWER.

		64 B packets			1460 B packets		
CW [MHz]	TX [dBm]	t_{max} [Mbps]	loss [%]	E_A [J/Mb]	t_{max} [Mbps]	loss [%]	E_A [J/Mb]
20	0	3.8	3.2	10	19.3	0.0	0.7
20	17	6.7	47.4	2.3	72.3	15.4	0.5
40	0	5.7	23.8	2.9	35.5	0.8	1.1
40	17	6.6	37.2	6.2	73.4	19.8	0.5

A. Detailed Analyses

In all our graphs in Figures 4–7, we have: (i) offered load on the horizontal axis (the configured values of offered bit rate, from our generated workload); (ii) used standard error bars, but in some cases, the error bars may be too small to be visible. In Figure 4, we show throughput, loss and E_A, for a 20 MHz channel, with the left column showing results measured at 0 dBm (1 mW) and the right column measured at 17 dBm (50 mW). Figure 5 shows the corresponding results for a 40 MHz channel. The graphs for E_A in Figures 4 and 5 show clearly the *effective energy usage envelopes*, the region between the lines plotted for small packets and large packets.

For throughput, upto 1 Mbps, we see very little difference between the different systems configurations (TX power, channel width), or workload (packet size, offered bit rate). However, beyond this, we see the expected changes: maximum TX power (17 dBm) and wider channel (40 MHz) give better throughput than minimum TX power (0 dBm) and the normal channel width (20 MHz). A counterintuitive observation is that in all our experiments with high TX power (which also equates to higher RSSI in our setup), higher throughput comes at the cost of a higher loss rate. However, for our energy usage metric, E_A, we see the largest impact is made by the adjustment of the application-level workload: increasing packet size and increasing data rate yield better energy usage.

We have made a comparative analysis of the 802.11n configurations with respect to both transmission power and channel width, which we will call a *delta* (Δ) analyses. For transmission power, in Figure 6: (i) Δ throughput was computed as the normalised relation of $throughput_{0dBm}/throughput_{17dBm}$; (ii) as the loss is already a normalised value, we have simply computed Δ loss as the difference of $loss_{0dBm} - loss_{17dBm}$; (iii) for energy usage, ΔE_A, was computed as the normalised value of E_{A0dBm}/E_{A17dBm}. For channel width, in Figure 7: (i) Δ throughput was computed as the normalised relation of $throughput_{20MHz}/throughput_{40MHz}$; (ii) as loss is already a normalised value Δ is the difference of $loss_{20MHz} - loss_{40MHz}$. For energy usage, ΔE_A was derived from the normalised relation of E_{A20MHz}/E_{A40MHz}. Here, again, we see that; (i) there is not much difference observed below 1 Mbps; (ii) the main differences occur due to the packet size and data rates of the offered loads.

Overall, below ∼1 Mbps, we see that there is little difference in performance with respect to loss and throughput across *all* the experiments. This applies to 20 MHz and 40 MHz channels; small packets and large packets; and to low and high transmission power. So, where application flows are below 1 Mbps, there may not be much benefit in performance by applying (self-)management actions in adjusting the packet size or data rate. However, we do see a significant difference (an order of magnitude) in effective energy usage for data rates values from low data rates up to 1 Mbps. This means that, for applications developers, the main incentive for applying (self-)management actions is going to be energy efficient as there is no loss (or gain) in performance.

Above 1 Mbps, we again see significant differences in effective energy usage (another order of magnitude from ∼1 Mbps to ∼25 Mbps), but the other performance parameters (loss and throughput in our measurements) vary greatly. In this case, a more complex (self-)management system, policies or mechanisms may be required to achieve lower effective energy usage without compromising performance.

So, from a management viewpoint, the system-wide effective energy usage and other performance parameters must be considered within any (self-)management functionality. In the next Section, we discuss how such trade-offs could be made for different classes of applications, based on our experiments.

IV. MANAGING ENERGY-AWARENESS IN APPLICATIONS

Based on our analyses in Section III, we can now determine management actions and interventions that are possible in order to enable energy efficient operation on a system-wide basis by actions at the application-level. Our results also tell us what is achievable: in Figure2, we have the mean effective energy usage envelope across all our experiments.

A vertical 'profile' of this graph, for example, gives us the margin of change that is possible for the same data rate but with different packet sizes: the lower line shows the best effective energy usage we can obtain.

A horizontal 'profile' of this graph shows us that, what adjustment is possible to the maximum data without increasing effective energy usage. Of course, this is not the complete picture: other performance issues, such as loss, may need to be considered, as shown in Figures 4 and 5.

Some applications will have greater scope for improving effective energy usage than others. Different approaches may be required. For instance, VoIP applications require real time data exchange and may be able to adapt their flow characteristics in when the network conditions permit. The same applies to ViIP applications, but as they normally do not require real-time data transfers they may be able to apply a greater magnitude of change to their flow characteristics and also use increased buffering/caching at the application layer. We see that E_A in file transfer applications also benefits from higher data rates.

A. Energy-aware self-adaptation for VoIP flows

VoIP applications use small packets to minimise impact of packet loss and to reduce end-to-end delay. However, we observe that use of small packets is not energy efficient. If network conditions permit, a VoIP application could alter its packet size, without changing its offered load, in order to improve E_A. For example, consider Figure 3, which depicts the delay budget for a VoIP application.

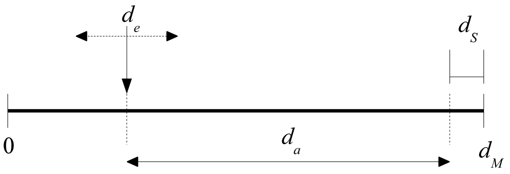

Fig. 3. Trading off delay for packet size in a VoIP transmitter. The maximum delay tolerable, d_M, is the upper bound. The application can estimate the current end-to-end delay, d_e, using reports from the receiver. Assuming some safety margin, d_S (includes receiver-side delay), the application can delay the packet transmission by d_a to create larger packets.

For a VoIP application to produce larger packets, it has to delay packet transmission more than for a smaller packet size, increasing the per-packet, end-to-end delay. Given an upper bound on the maximum delay for the application, d_M, there may be an acceptable delay, d_a, that could be introduced by the application. This delay, d_a, could be evaluated by taking measurements of the current end-to-end delay (d_e, variable), using existing means (e.g. RTCP Reports), and allowing some safety margin (d_S, for receiver delays such as decoding and playout buffering for jitter smoothing). Additionally, the loss of a larger packet could result in greater impact at the receiver. Of course, such adaptation would also depend on the capabilities of the audio codec in use, as different codecs have different encoding/decoding delay and tolerance to loss. e.g. a G.711/PCM-based codec may be more amenable to such techniques than a CELP-based codec. The simple example above shows only delay, but other factors may need to be considered. Overall, the application would need to assess current network and end-system conditions against its own operating modes in order to make appropriate adjustments, e.g. assess current packet loss experienced by the application, and current power usage – different codecs may have different impact on power consumption in an end-system, for example as described in [10]. Such adaptation could also be integrated with congestion control mechanisms (e.g. DCCP[8]), or as part of an autonomic management policy (e.g. with the management framework introduced in [16]). Similar considerations would apply to real-time video, with the added complexity of video coding/framing.

B. Energy-efficient video streaming

For ViIP (streamed, non-real-time video), if client-side buffering is used to compensate for loss, it may be possible to use a similar approach as in Section IV-A, but with less aggressive constraints, as non-real time transfers are more tolerable to end-to-end delay and loss. In this case, dynamic codec selection may be used instead of packet size adaptation in order to change data rates, or, where possible the use of modern scaleable codecs such as H.264 AVC. In the limit, with large buffering at the client-side, the streaming of a video file can be approximated to the case of file transfer (see below). If we consider the results of the E_A values in Figures 4 and 5, we see that the emulated youtube traffic has E_A values that are much higher than our emulations of file transfers using HTTP. Again, the energy cost of the video codec must be considered.

C. Green caching for file transfer applications

File transfer applications (e.g. HTTP) already operate in fairly energy efficient manner, according to our measurements. So, there is a lower margin for improving effective energy usage by adapting their flow characteristics. However, we also observe that increasing data rates also reduces effective energy usage. Our measurements for a HTTP download locally (Intranet) and for a remote server (Internet) show that the local download at a higher data rate was more energy efficient – by an order of magnitude in our particular measurements, as shown in Section III. So, if content is cached locally, this may improve download data rates and energy usage. Of course, locally cached content also avoids the energy overhead of using the network resources for fetching the content from the remote source. Although this may be hard to assess quantitatively, the intuition is that it will be more energy

---

[8] http://www.erg.abdn.ac.uk/~gerrit/dccp/apps/

efficient. As site-wide caches are already available and used in widely today, this offers an easy path to more energy efficient file transfers.

Now, this may be seen as a cost trade-off: could monetary savings in energy usage and lower volumes of downstream traffic due to local caching be sufficient to offset the cost of the installation and maintenance of the cache? Developing appropriate models to investigate this is future work. Such caching may not be appropriate for all content or all content providers. For example, the youtube business model requires visits to a youtube server so that the number of views of content can be recorded. This is an engineering challenge, again, suitable for future work.

V. RELATED WORK

The authors' previous work [11] compares energy efficiency in IEEE 802.11a and IEEE 802.11n at 5 GHz. It presents an initial evaluation of the *effective energy usage envelope* and provides the basic model for this study. The key new contributions in this paper are to consider the *dynamics* of the envelope with: (i) lower transmission power, 0 dBm (only 17 dBm previously); (ii) a 40 MHz channel (only 20 MHz previously); (iii) higher data rates, up to 90 Mbps (only up to 25 Mbps previously); (iv) emulated data transfer with HTTP via Internet and Intranet (previously only Skype and YouTube). As well as work already listed in the Introduction, we provide below a non-exhaustive summary of other relevant work.

In [17], the authors examine 802.11n energy efficiency and conclude that transmissions with larger packets and higher data rates are more energy efficient than those using smaller packet sizes and lower data rates. However, their study measures the power consumption directly at the wireless NIC, so they do not capture system-wide effects. Additionally, their consideration of bit rates is by looking at the modulation and coding scheme (MCS) that is selected by the WLAN driver, while we consider the data rate that is *measured* at the application-level.

In [18], the authors study energy efficiency and propose their own extension to the 802.11 MAC protocol in order to improve energy efficiency. Alternatively, in [19], the authors consider an analytical model to evaluate energy efficiency. They also propose an enhancement to 802.11, not at the protocol level, but by allowing non-transmitting or receiving NICs to remain in sleep mode for the duration of any ongoing transmissions that are not of interest to those nodes. We have not considered such work in our study, as our objective was to understand typical energy efficiency for off-the-shelf equipment with out-of-the-box configuration.

As we wished to consider off-the-shelf equipment, with out-of-the-box configurations, we have for this study excluded use of 802.11n features which are not normally turned on by default, might be optional for implementation by vendors, are vendor specific, or would require specific expert knowledge by the user in order to configure those features for use. Such features include 802.11 Power Save Mode (PSM) [2], Unscheduled Automatic Power Save Delivery (U-APSD) [3], WMM Power Save (WMM-PS) [4], Dynamic MIMO Power Save [5], and Wake-on-Wireless [6]. A survey of energy efficiency through MAC layer techniques is presented in [20].

There are recent studies on performance, e.g. [21], [22], [23], but none of these consider energy usage. Our performance results are in general agreement with those studies.

VI. CONCLUSION

We have performed an empirical investigation of IEEE 802.11n at 5 GHz in order to investigate its *energy usage dynamics*, including how the *effective energy usage envelope* is effected by changes in transmission power and channel usage. We find that, as might be expected, transmission power and channel width do effect performance, such as throughput and loss. However, they have relatively little impact on application-level energy usage. Our key observation is that *we can gain a better appreciation of energy usage by considering the amortisation of energy usage across the system as a whole*.

The greatest impact on effective energy usage, E_A, results from the use of large packets and higher transmission rates: the energy usage of the system is amortised over the greater number of bytes that are transmitted. We see changes of two orders of magnitude in effective energy usage with changes in data rates, and significant changes in energy usage for the same data rate but with different packet sizes.

This means that *there is great potential for application-level (self-)adaptation to achieve greater energy efficiencies than is currently achieved for IEEE 802.11n at 5 GHz, by adjusting packet flow construction and packet transmission*. We also provide examples of how our findings can be exploited in order to allow applications to trade off performance against energy usage, but different mechanisms may apply to different classes of applications. Although our experiments were carried out for a WLAN cell, as we are observing the amortisation of the energy/transmission costs of the end-system, the results are also found to be similar for wired Ethernet [11].

This study is a first step towards understanding the dynamics of real IEEE 802.11 systems with respect to building energy-aware (self-)adaptive applications. Future extensions of this work include (but are not limited to) examining multiple clients in a cell; considering 2.4 GHz, e.g. 802.11b and 802.11g; comparing application-specific and OS-specific energy issues; and of course, considering the nature of the application-level self-adaptation that can be realised based on our new understanding of IEEE 8021.11 energy usage dynamics. User studies would also be required in order to determine the efficacy and impact on quality of experience (QoE) of such adaptive techniques for multimedia applications. Such adaptation of traffic flows would also impact network management techniques based on packet analysis.

ACKNOWLEDGEMENTS

Lito Kriara (University of Edinburgh, UK) provided feedback on this work. Prof Krishna M. Sivalingam and Narendran Krishnan (IIT Madras, India) provided discussion and motivational background. This work has been supported by the EPSRC-funded project IU-ATC (http://www.iu-atc.com/).

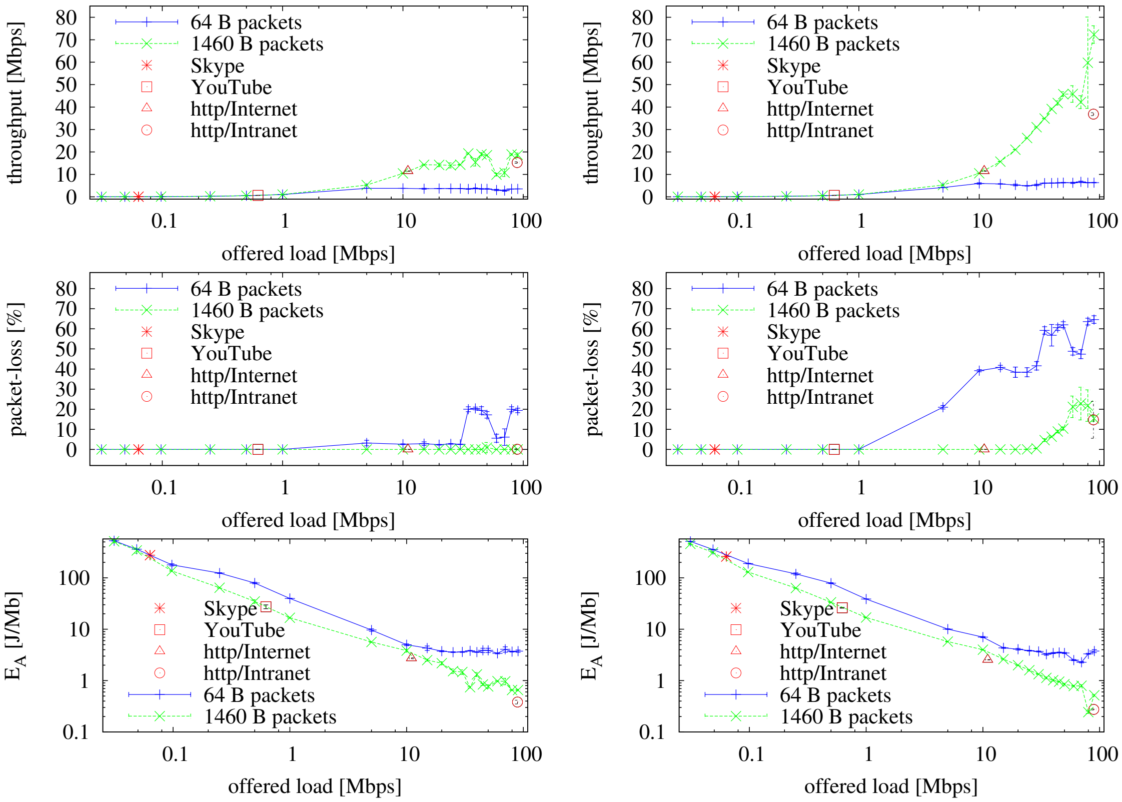

Fig. 4. IEEE 802.11n 20 MHz Channel - 0 dBm (left column) and 17 dBm (right column) TX power.

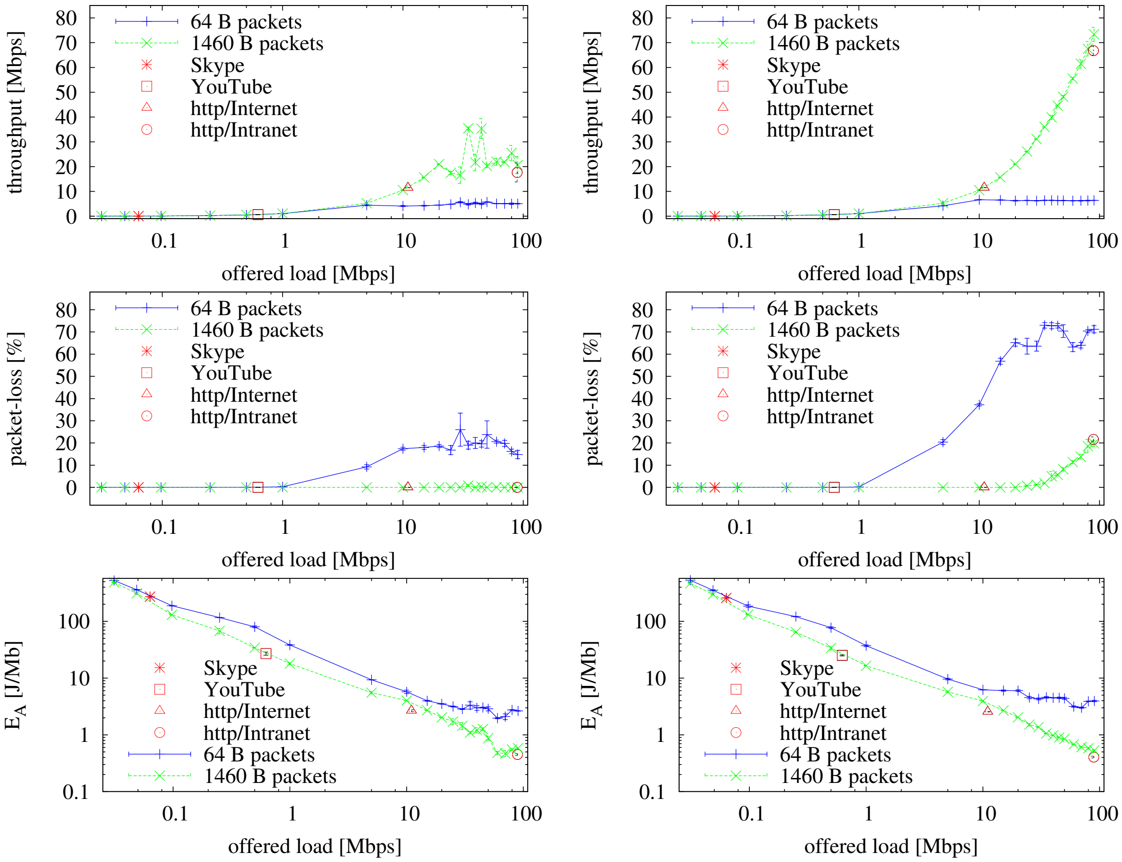

Fig. 5. IEEE 802.11n 40 MHz Channel. 0 dBm (left column) and 17 dBm (right column) TX power.

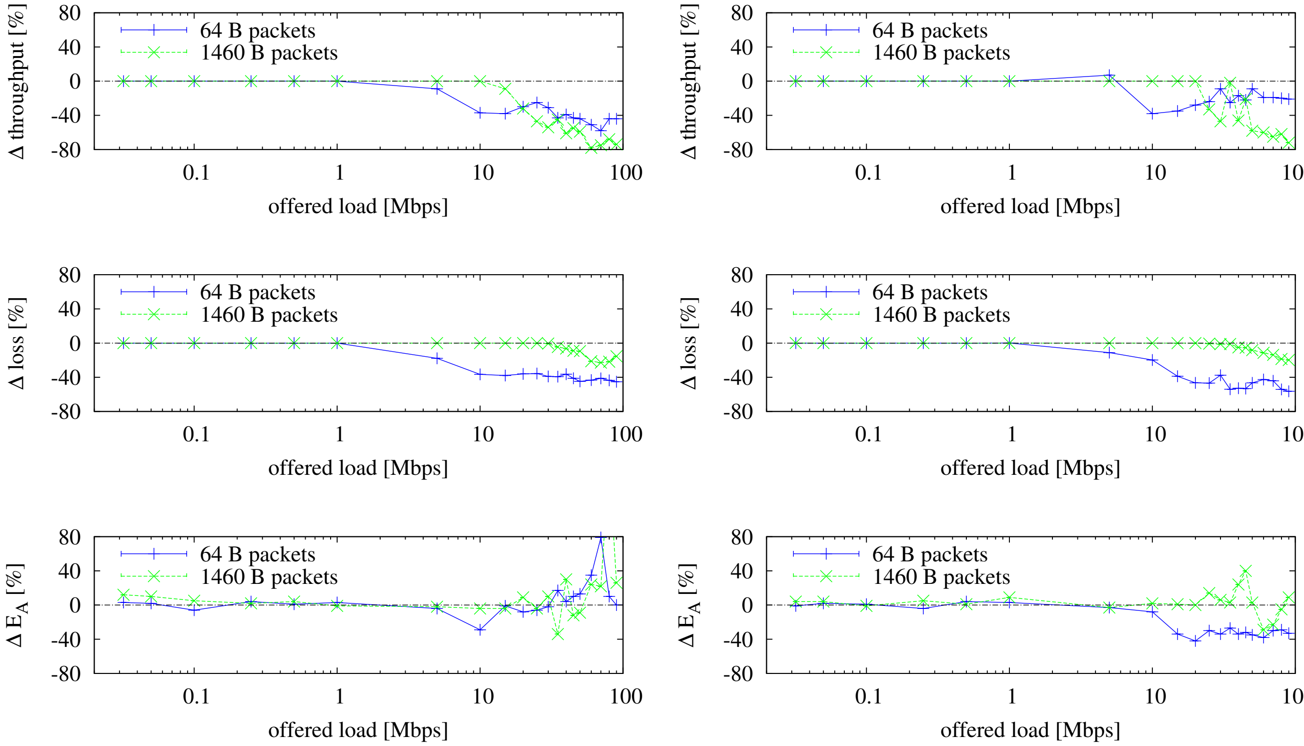

Fig. 6. Differences in the main observables depending on TX power (0dBm - 17 dBm). 20 MHz channel (left column) and 40 MHz channel (right column). Horizontal zero line is a visual aid. Positive values indicate where 0dBm TX power has higher values. Application-specific workloads omitted for readability

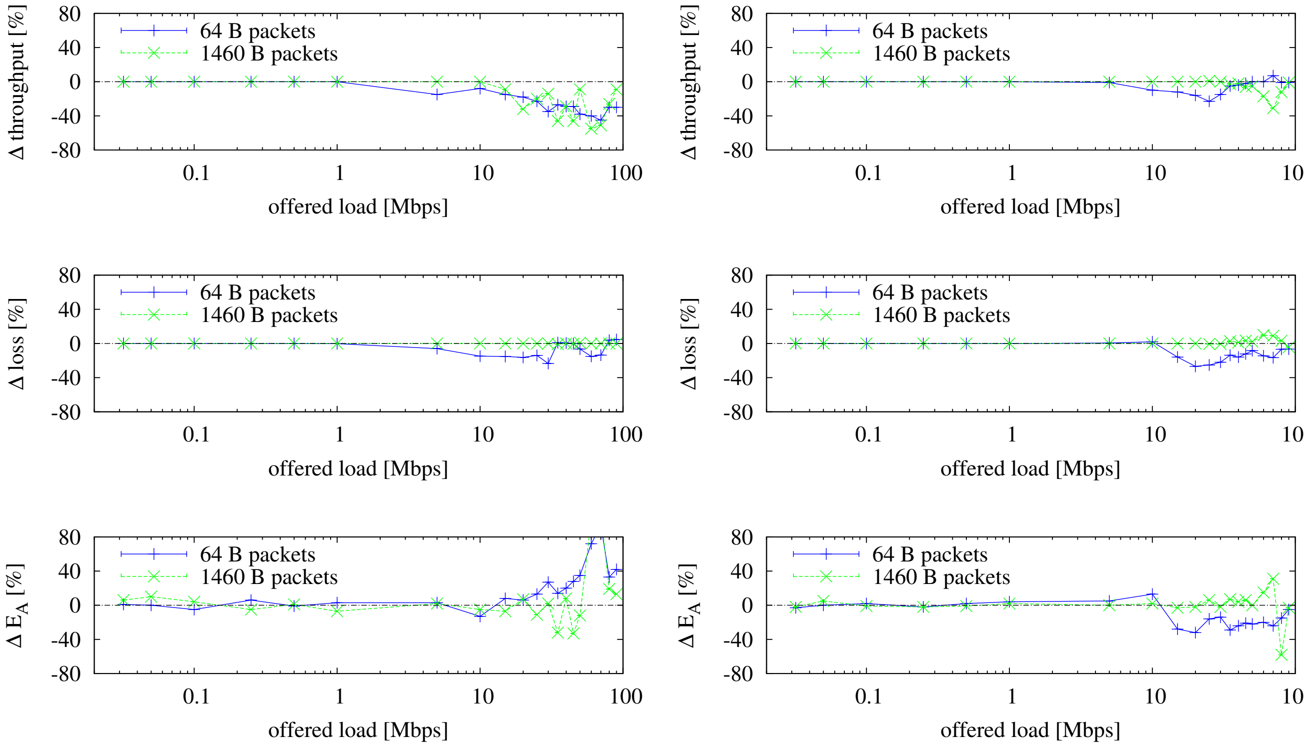

Fig. 7. Differences in the main observables depending on channel width (20Mhz - 40MHz). 0 dBm TX power (left column) and 17 dBm (right column). Horizontal zero line is a visual aid. Positive values indicate where 20MHz channel has higher values. Application-specific workloads omitted for readability

REFERENCES

[1] "SMART2020: Enabling the low carbon economy in the information age." The Climate Group, Tech. Rep., 2008. [Online]. Available: http://www.smart2020.org/

[2] D. Qiao and K. Shin, "Smart power-saving mode for IEEE 802.11 wireless LANs," in *Proc. INFOCOM 2005*, vol. 3, Mar 2005, pp. 1573–1583.

[3] H. Liu and Y. Wang, "An Implementation Scheme of Power Saving Mechanism for IEEE 802.11e," in *Computer Science and Software Engineering, 2008 Intl. Conf. on*, vol. 1, Dec 2008, pp. 1178–1181.

[4] B. Alfonsi, "In brief: Wi-Fi certification program aims to boost battery life," *IEEE Distributed Systems Online*, vol. 7, no. 1, Jan 2006.

[5] H. Kim, C.-B. Chae, G. de Veciana, and R. Heath, "Energy-efficient adaptive MIMO systems leveraging dynamic spare capacity," in *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*, Mar 2008, pp. 68–73.

[6] N. Mishra, K. Chebrolu, B. Raman, and A. Pathak, "Wake-on-WLAN," in *Proceedings of the 15th international conference on World Wide Web*. New York, NY, USA: ACM, 2006, pp. 761–769.

[7] G. Bernardi, P. Buneman, and M. K. Marina, "Tegola Tiered Mesh Network Testbed in Rural Scotland," in *Proc. ACM Mobicom 2008, Workshop on "Wireless Networks and Systems for Developing Regions" (WiNS-DR)*, 2008.

[8] S. Baek and B. D. Choi, "Performance analysis of power save mode in IEEE 802.11 infrastructure WLAN," in *Telecommunications, 2008. ICT 2008. Intl Conf. on*, Jun 2008, pp. 1–4.

[9] T. Jun, Z. Ying-jiang, W. An-qing, Y. Zhi-wei, and J. Ming-bo, "A research of wlan packet aggregation," in *Education Technology and Computer Science (ETCS), 2010 Second International Workshop on*, vol. 2, Mar 2010, pp. 480–483.

[10] S. N. Bhatti and G. Knight, "Enabling QoS adaptation decisions for Internet applications," *Computer Networks*, vol. 31, pp. 669–692, 1999.

[11] M. Tauber, S. N. Bhatti, and Y. Yu, "Application Level Energy and Performance Measurements in a Wireless LAN," in *Proc. GreenCom2011: IEEE/ACM Intl. Conf. on Green Computing & Comms.*, Aug 2011.

[12] K.-T. Chen, C.-Y. Huang, P. Huang, and C.-L. Lei, "Quantifying Skype User Satisfaction," in *Proc. SIGCOMM 2006*, Aug 2006.

[13] D. Bonfiglio, M. Mellia, and M. Meo, "Revealing Skype Traffic: When Randomness Plays with You," in *Proc. SIGCOMM 2007*, Aug 2007.

[14] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Watch Global, Cache Local: YouTube Network Traffic at a Campus Network - Measurements and Implications," in *Proc. IEEE MMCN 2008*, 2008.

[15] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "Youtube traffic characterization: a view from the edge," in *Proc. IMC 2007*, 2007.

[16] M. Tauber, G. Kirby, and A. Dearle, "Self-Adaptation Applied to Peer-Set Maintenance in Chord via a Generic Autonomic Management Framework," in *Self-Adaptive Networks (SAN) Workshop, IEEE Intl. Conf. on Self-Adaptive and Self-Organizing Systems (SASO2010)*, 2010.

[17] D. Halperin, B. Greensteiny, A. Shethy, and D. Wetherall, "Demystifying 802.11n power consumption," in *Proc. 2010 International Conference on Power Aware Computing and Systems*. Berkeley, CA, USA: USENIX Association, 2010.

[18] S.-L. Wu and P.-C. Tseng, "An Energy Efficient MAC Protocol for IEEE 802.11 WLANs," *Communication Networks and Services Research, Annual Conference on*, vol. 0, pp. 137–145, 2004.

[19] M. Ergen and P. Varaiya, "Decomposition of Energy Consumption in IEEE 802.11," in *IEEE Intl. Conference on Communications, 2007. ICC '07*, 2007, pp. 403–408.

[20] S.-L. Tsao and C.-H. Huang, "A survey of energy efficient MAC protocols for IEEE 802.11 WLAN," *Computer Communications*, vol. 34, no. 1, pp. 54 – 67, 2011.

[21] S. Sendra, P. Fernandez, C. Turro, and J. Lloret, "IEEE 802.11a/b/g/n Indoor Coverage and Performance Comparison," in *6th International Conference on Wireless and Mobile Communications (ICWMC)*, 2010.

[22] J. Pacheco de Carvalho, H. Veiga, N. Marques, C. Pacheco, and A. Reis, "A contribution to performance measurements of IEEE 802.11 a, b, g laboratory WEP point-to-point links using TCP, UDP and FTP," in *International Conference on Applied Electronics (AE 2010)*, 2010.

[23] E. Gelal, K. Pelechrinis, I. Broustis, S. Krishnamurhty, S. Mohammed, A. Chockalingam, and S. Kasera, "On the Impact of MIMO Diversity on Higher Layer Performance," in *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, 2010, pp. 764–773.