WLAN Power Save by Header Compression and Packet Overhearing Reduction

David Levy[*]
Institute of Applied Informatics
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3, 842 16 Bratislava, Slovakia
david.levy@stuba.sk

Abstract

Cellular, laptop, tablet and handheld devices support WLAN technology. In addition to the spectral efficiency and security issues, power consumption is a vital issue in making the usage of these devices widespread. Efforts are underway in improving power conservation in those devices and thus increasing the duration between recharging the battery. Power consumption is mainly due to the transmission of WLAN packets and the receiver listening and receiving packets. Internet packets contain several headers for routing required in a complex network. Those networks are mostly wired networks using TCP/IP protocol. The WLAN transmitter is encapsulating all the TCP/IP headers and adding its own WLAN header. In some cases, the combined headers length represents a significant portion of the total frame length.

This paper focuses on reducing the combined headers length with different compression methods. Smaller headers imply smaller packets which take less time to transmit and therefore activate the power hungry WLAN transmitter for a shorter time. The receiver on time is also reduced using overhearing avoidance techniques. The performance of those new methods is evaluated using both simulation and analytical models. Both the simulation and the analytical WLAN power estimation were developed for this paper.

The simulation and the analytical models concord and show that if those methods are combined, they can provide up to 30% power consumption reduction for high load and many active stations present on the same channel.

[*]Recommended by thesis supervisor: Assoc. Prof. Ivan Kotuliak. Defended at Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava on August 22, 2014.

Levy, D. WLAN Power Save by Header Compression and Packet Overhearing Reduction. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 6, No. 3 (2014) 9-18

Categories and Subject Descriptors

C.2.1 [**Computer Systems Organization**]: Computer-Communication Networks—*Network Architecture and Design, Wireless communication*; C.2.5 [**Computer Systems Organization**]: Computer-Communication Networks—*Local and Wide-Area Networks, Access schemes*; C.4 [**Computer Systems Organization**]: Performance of Systems; E.4 [**Data**]: Coding and Information Theory—*Data compaction and compression*; I.6.4 [**Computing Methodologies**]: Simulation and modeling—*Model Validation and Analysis*

Keywords

802.11, WLAN, power consumption, analytical model, header compression, unknown header, packet overhearing, throughput

1. Introduction

There are existing improvements to the initially released IEEE 802.11 protocol , with mandatory features only. The WLAN device, also called STA (Station), stays in a less power consuming state when it is assumed that there is no relevant data that is received from the network. Unfortunately these improvements save power for devices that have a rather low average throughput. This is the case for voice applications for example.

As the applications demand for throughput on hand-held devices is exploding (wireless display, gaming, video over IP), the existing methods cannot manage to save enough power anymore. There is a need to extend battery life of devices which, for a long period of time (several hours), have a sustained high throughput.

The goal in this paper is to save power at a STA that has high throughput, as this is not dealt with in the existing power save protocols. The environment is assumed such that also the other STAs on the channel have a rather high throughput.

Two complementing ideas will be developed in this paper:

- Header compression,
- Packet overhearing reduction.

Header compression is most efficient on short packets while packet overhearing avoidance is most effective for longer packets.

For header compression, the idea to reduce the time the STA stays in transmit or receive state is to reduce the frame size by reducing the header size. The header of a frame is, in fact, a series of encapsulated headers. Each header represents the overhead required by a specific protocol. The header represents therefore a substantial part of a typical frame. A typical encapsulation of a part of an Internet page, has the following headers: WLAN header, LLC header, IP header, TCP header and HTTP header. The frame overhead can range from 66 bytes for Voice over IP applications to more than 100 bytes for Internet applications.

The existing header compression methods compress only two known protocols concurrently. The novel compression scheme proposed in this paper will have a lower compression performance but will cover all the Open Systems Interconnection reference model layers, from Physical to Application. The compression scheme does not need to understand the upper layer protocols in order to compress the headers. Only the WLAN header is compressed with protocol knowledge for packet re-ordering and correct channel access.

In order to follow the CSMA/CA protocol which is part of the standard, all STAs stay in the listen state and hear all traffic, including frames that are not addressed to them. Not addressed frames are discarded after they were fully received. A study of Biswas in [3] shows that in a high traffic network, more than 30% of a device energy is spent in receiving packets that are not addressed to it. Therefore, emphasizing on reducing this energy can save substantial energy, overall, for the device. The novel idea in packet overhearing reduction is to reduce the time a STA stays in receive state for acquiring a frame that is not addressed to it.

In order to estimate the performance of the proposed methods, a novel analytical model of a WLAN STA power consumption was developed. The analytical model is derived from Bianchi's throughput analytical model of a WLAN STA developed in [2].

The organization of this paper is as follows: Section 2 will mention existing methods to estimate and save power in WLAN. Section 3 will describe the novel power consumption analytical model. Section 4 will describe the proposed header compression algorithms. Section 5 will explain the power overhearing reduction method. Section 6 will evaluate the performance of the two proposed methods. Section 7 will conclude this paper.

2. Prior Art

This section will mention Prior Art of WLAN power save methods, header compression methods, packet overhearing reduction and throughput analytical models.

There are several amendments to the 802.11 standard that provide mechanism to save power. They are well described in [9]. Those mechanisms try to have the STA stay longer in sleep state instead of listen state. Since the power consumption in the sleep state is much lower than in the listen state , if implemented correctly, these protocols provide a substantial battery life extension. Unfortunately these protocols save power for devices that have a rather low average throughput. In case of sustained high throughput, the device would have to stay anyway in idle state to be able to continuously transmit and/or receive following the CSMA/CA protocol.

The existing header compressions were developed mainly to increase the performance of the IP flows. The older compression methods [7] and [5] used the simple delta compression, sending only the difference in the value of the changing fields, to minimize the number of bits sent. The TCP recovery mechanism was used to recover from channel errors. Newer compression schemes do not use the TCP, anymore, to recover from error conditions. Instead, a dedicated feedback mechanism was defined and allowed compression of other headers such as UDP and RTP. The last development was the ROHC [4] which compresses more types of headers and offers also feedback mechanism for error recovery. ROHC defines some compression algorithms dedicated to specific headers to achieve very high compression efficiency. The compression methods compress only two known protocols concurrently. Unknown headers cannot be compressed and headers on other layers cannot be compressed either.

In order to cover all OSI layer headers, a general compression scheme shall be used, instead of a dedicated method for every layer. A comparison of several general compression algorithms was done in [8]. The non-adaptive Huffmann encoding [6] cannot be used as the headers are unknown. The adaptive Huffmann encoding requires too much processing time compared to the throughput of WLAN. The dictionary methods (as the Lempel-Ziv-Welch) or [12] are inefficient in the first kilo byte of the compressed message, but the headers field we need to compress is less than 200 bytes. The Run-Length Encoding (RLE) performs a lossless compression of input data based on sequences of identical values (runs). The algorithm is quite easy: each run, instead of being represented explicitly, is translated by the encoding algorithm in a pair (l, v) where l is the length of the run and v is the value of the run elements. The longer the run in the sequence to be compressed, the better is the compression ratio. An enhanced RLE was proposed in [1].

The WLAN standard proposes several methods to reduce device power consumption: Legacy power save mode, unscheduled Automatic Power Save Delivery and Power Save Multi-Poll. Other proposed methods go in the same direction to save power for a station which has low traffic [11], [10] and [13]. In all cases, low traffic is required to be able to reduce power consumption.

Biswas proposes, in [3], a way to avoid packet overhearing, relaying on the availability of the Clear To Send (CTS) packet. The address of the destination node in the CTS packet is checked. If it is not the one of the device, then the STA is forced to sleep state for the duration appearing in the CTS packet duration field. This corresponds to the duration of the immediately following data packet. The CTS mechanism was used for backwards compatibility and for hidden node avoidance. But in modern networks there is no need for backwards compatibility anymore and the receivers have much higher sensitivity than the transmitter, sharply reducing the effect of the hidden node. The CTS frame is seldom used as it reduces the system throughput.

In [2], Bianchi developed the WLAN throughput analytical model. His approach revolutionized the analytic study of networks employing this protocol. His model studies the behavior of a single STA with a discrete time Markov chain. He obtains the probability of transmission of this STA and the probability of collision in steady state. The study is then generalized to all the STAs on the channel, to obtain the overall channel throughput.

3. Analytical Model

This section describes how the STA average power consumption was derived from Bianchi's model. The analytical model is validated with comparison to a WLAN Matlab model, which I also developed.

3.1 Derive power consumption from throughput model

The average slot duration, σ_{avg}, is defined in (1). The terms (denominator of the normalized throughput obtained by Bianchi) are the successful transfer time, failed transfer time due to collision and listen time respectively. Unless clearly defined, all parameters appearing in this section are identical to those defined in [2].

$$\sigma_{avg} = (1 - p_{tr})\sigma + p_{tr}p_sT_s + p_{tr}(1 - p_s)T_c \qquad (1)$$

The fraction of time the network spends for a successful transfer, failed transfer and decrementing the back-off (listen time), is defined in equations (2), (3) and (4), respectively.

$$\sigma_s = \frac{p_{tr}p_sT_s}{\sigma_{avg}} \qquad (2)$$

$$\sigma_c = \frac{p_{tr}(1 - p_s)T_c}{\sigma_{avg}} \qquad (3)$$

$$\sigma_{bo} = \frac{(1 - p_{tr})\sigma}{\sigma_{avg}} \qquad (4)$$

These equations need to be translated to fractions of time during which a single STA is in transmit, receive and listen states in order to estimate the average power consumption of a single STA.

The fraction of time a STA is in transmit state, τ_{tx}, is the time it spends to transmit successful packets, colliding packets and Acknowledgments to packets it successfully received. The fraction of time a STA is in receive state, τ_{rx}, is the time it spends to receive successful packets, Acknowledgments and colliding packets. The fraction of time a STA is in listen state, τ_{idle}, is the time it spends to sense the channel (back-off counter decrement), the inter-frame times during successful packet reception and the DIFS time (defined in 802.11 standard) after colliding packets. All those time fractions are function of σ_s, σ_c, σ_{bo}, the header length, the payload length, the number of active STAs on the channel and the time required to transmit successful and colliding packets (including eventual Acknowledgment and Inter Frame Spaces, as defined in 802.11).

The fraction of time in each state can now be used to calculate the average power consumption of a STA:

$$P_{avg} = \tau_{idle}P_{idle} + \tau_{tx}P_{tx} + \tau_{rx}P_{rx}, \qquad (5)$$

with P_{idle}, the power consumption of a STA in listen state, P_{tx}, the power consumption of a STA in transmit state and P_{rx}, the power consumption in receive state. Those powers are usually provided by the device manufacturer in their data sheets.

3.2 Validation of the analytical model

The power consumption analytical model developed in section 3.1 needs to be validated. For this purpose I developed internally (in Matlab) a simulation that emulates a network where all the stations follow the CSMA/CA protocol, as defined in 802.11. The correct behavior of the simulator itself was checked by comparing the simulator normalized throughput to the analytical model defined in [2]. Once the throughput is correct, it is guaranteed that the behavior of all the stations on the simulator access the media as defined by the CSMA/CA protocol. Each STA is correctly switching power states (transmit, receive and listen). The power consumption estimation for each station in the simulation is then simply accumulating the energy based on the station power state. The power consumption analytical model was validated by comparing the results to this WLAN network simulator. There were only minor differences, validating the power estimation analytical model.

The developed analytic and Matlab models will be used in following chapters to estimate the performance of the proposed power save methods.

4. Header Compression Algorithm

In this section, I focus on decreasing header size by proposing generic compression algorithms.

In wireless networks, the channel introduces many errors (up to a PER of 10%). If those errors are undetected at the link layer, corrupted frames can be forwarded to the higher layers. When header compression is performed, corrupted headers could cause misinterpretation of data content at the receiver. A single corrupted frame forwarded at the higher layers can therefore cause the loss of several subsequent frames. To avoid this loss, as many errors as possible shall be detected. Due to the number of error bursts generated by the Viterbi decoder, the 32-bit WLAN FCS cannot be trusted for detecting all practical errors. A mechanism needs to be developed to avoid transfers of undetected corrupted packets to the higher layer.

I will use a 16 bit CRC to protect the compressed header alone. The header that can be compressed will be limited to 200 bytes to reduce the probability of multiple bursts on the protected compressed header. The average distance between error bursts, decreases exponentially with the SNR. Compression will be allowed only on links that are assumed to have a good SNR. The receiver first checks the FCS. In case of an invalid FCS the frame is rejected and no Acknowledgment (ACK) frame is sent. If the 16 bit CRC detects error that was not detected by the FCS, then the receiver shall

1. stop sending ACKs to subsequent frames until an uncompressed packet is received,
2. not forward any frame to the upper layer until an uncompressed frame is received.

The general flow is modified to include compression and decompression.

As shown in Figure 1, the compression has to be split to three stages. First, all layers above the WLAN MAC layer are compressed before encryption. This is needed because the encrypted data looses the repetitive pattern that is visible between consecutive frames. After the MAC layer is added, the second compression stage can be executed on it. Finally, the Duration ID field which is part of the MAC header has to be modified to indicate the compressed packet duration.

The decompression is split to two stages, one for the MAC header and one for the rest of the headers. The frame reordering and decryption must take place before the second decompression stage.

I limit compression to WLAN data frames (excluding control and management frames) in an Infrastructure network. Each WLAN header field is compressed in a different way to optimize compression. By this I achieve close to 50% compression of the WLAN MAC header.

The transmitter separates flows based on the Access Category (AC). Each AC queue is treated independently. The compression consists in comparing the previously successfully transmitted frame with the frame to be sent on the same AC queue. The compressor calculates the difference between the two frames for every byte. The resulting difference is then compressed. The decompressor decompresses the header and rebuilds the newly received frame by adding to it the previously successfully received frame. Two problems need to be considered:

1. What is the optimum compression method, taking into account the limited resources and processing time?
2. The length of the generalized header is unknown, as the protocols are unknown. It is clear, that at a certain point, the application payload data will appear in the frame. From this point on, all bytes will differ, compared to the previous frame. The effect may be an increase in length, instead of the wished decrease, due to many differences between two consecutive frames. What is the optimum number of bytes to compress?

In order to decompress, the compressor must tell the decompressor the length of the compression code. This is pre-pended in the one byte Length field. The 16 bit CRC covers all compressed content, including the compressed WLAN header. The frame format is shown in Figure 2, where the untouched part of the frame is grayed out.

The Bit Stuffed (BS) RLE that was proposed in [1] is compared to a newly proposed method called Fixed Asymmetric (FA) RLE. The FA RLE is a simplification of the BS RLE and biased to compress better the case where no change happened between two consecutive frames. All runs are forced to have the same length. Then it is not more needed to transmit the number of repetitions, since it is *fixed* and known by the receiver.

Since the length of the unknown header is also unknown, algorithms to detect the boundary between the end of the header and the beginning of the payload are also created. The idea is to find the number of bytes to compress that yields the shortest compressed header. The algorithm calculates the header reduction for each additionally compressed header byte, using the BS RLE or the FA RLE,

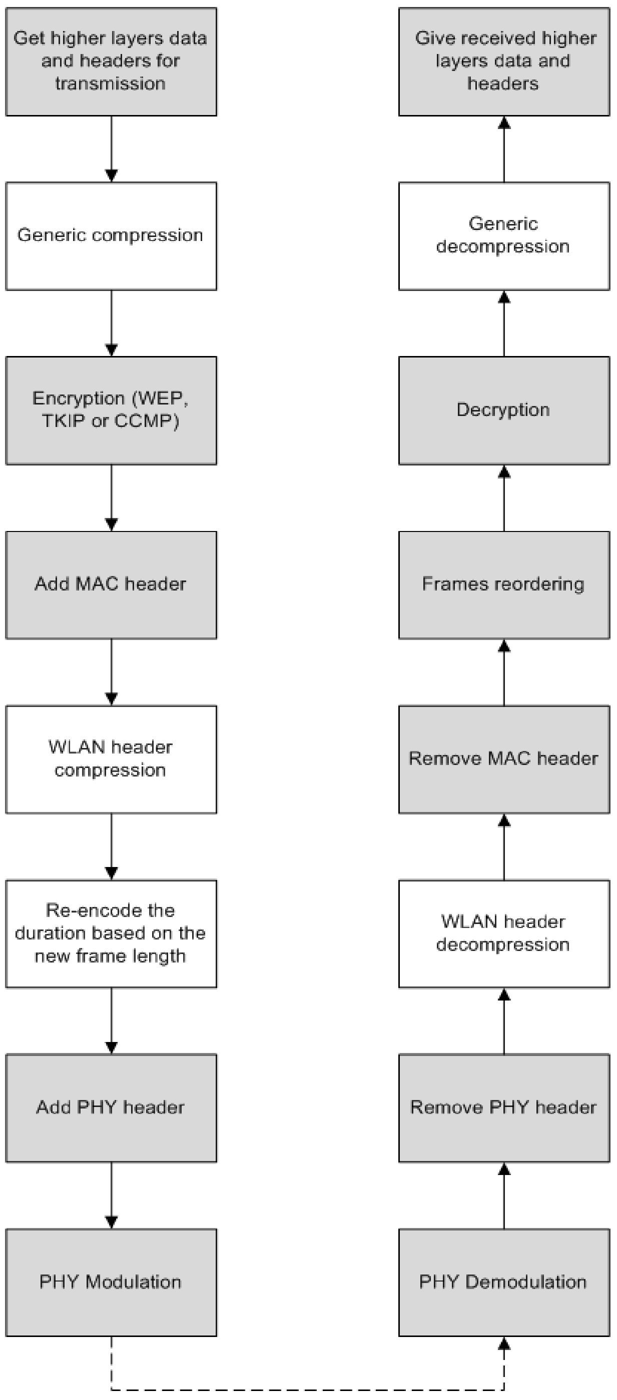
Get higher layers data and headers for transmission
Generic compression
Encryption (WEP, TKIP or CCMP)
Add MAC header
WLAN header compression
Re-encode the duration based on the new frame length
Add PHY header
PHY Modulation
Give received higher layers data and headers
Generic decompression
Decryption
Frames reordering
Remove MAC header
WLAN header decompression
Remove PHY header
PHY Demodulation

Figure 1: Flow with compression

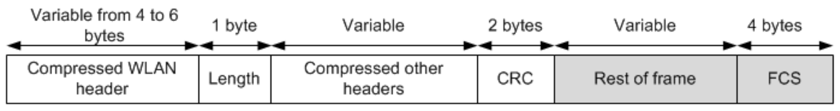

Figure 2: Compressed frame format

therefore it is called *Header Reduction* algorithm. I then obtain a graph that shows the header reduction against the number of compressed bytes (starting from the end of the WLAN MAC header). The peak in this graph is the optimal number of bytes to compress.

At the end of the unknown header length, the difference between consecutive frames will suddenly grow. Therefore, a simpler approach to find the boundary is to perform a moving average of the binary difference. The binary difference between consecutive frames for every byte is 1 if the bytes are different and 0 if they are equal. If the moving average is above a threshold, then the header-data boundary was reached. This algorithm is called the *End of Header Detection* algorithm.

Additionally, I use analytical models for performance evaluation. I developed an analytical model in the simple case, where a single STA is active. This model calculates both the energy and the throughput gained by header compression. The second analytical model for more active STAs is a modified version of the model presented in section 3.1. The only difference is that the header length for packets that are transmitted first time right (not colliding) is reduced.

5. Packet Overhearing Reduction

This section describes a method to reduce the WLAN inherent packet overhearing without the usage of the Request to Send (RTS)/CTS mechanism. For this purpose I introduce a new intermediate state called *semi-doze* state. In this state, a STA consumes less power than in listen/idle state, because it has some of its analog components switch off. But the slowest analog components are not switched off, so that it is possible to return to receive state in micro-seconds (and not milliseconds when waking up from doze state).

The proposed method consists in:

1. detecting early in packet reception that the frame is addressed to another device,
2. calculating the remaining duration of the packet. If this duration is longer than the time it takes for the STA to transition from receive to semi-doze state and back to receive state, then
3. changing to semi-doze state, and,
4. scheduling to order transition back to the receive state such that the STA will finish its transition shortly before the end of the packet.

Like in the method proposed in [3], the gain in power is P_{rx} - $P_{semi-doze}$, where $P_{semi-doze}$ is the power consumption in semi-doze state. The duration of this power gain is not the complete data packet τ_p, as one needs to take into consideration the transitions of the STA from state to state, as shown in Figure 3. Therefore, I talk of packet overhearing *reduction*. The duration of the energy gain is τ_g:

$$\tau_g = \tau_p - \tau_{res} \quad (6)$$

where τ_{res} is the duration of residual overhearing. It can be partitioned to $\tau_{res} = \tau_d + \tau_{sd} + \tau_r + \tau_m$ where τ_d is the time it takes the receiver to detect that the packet is addressed to another STA, τ_{sd} and τ_r are the transition time to semi-doze and and back to receive state, respectively. τ_m is a margin used to take into account the eventual STA to AP clock phase shift.

In order to check the performance of this method, the analytical model defined in section 3.1 is modified. It takes into account also the time a STA is in semi-doze state. Another analytical model is developed for the simple case where only the AP transmits to two STAs. The latter shows an asymptotic gain (for very long frames) as defined in equation 7.

$$G_{max} = \frac{1}{2}\frac{P_{semi-doze}}{P_{rx}} \quad (7)$$

6. Evaluation of proposed methods

The objective of this section is to evaluate the gains achieved by header compression and packet overhearing avoidance. The methods to achieve this objective are both analytical and by simulations. I will first measure on real captured frames the average header reduction that can be achieved by Header compression. This will be used to estimate the energy consumption and throughput gains when the frames have the reduced length. The packet overhearing reduction energy consumption gain will also be estimated. Finally, both methods will be combined to figure out how each of them contributes to save energy and how much can be saved in total. The performance will be estimated when varying the key parameters, as the payload length, the PHY rate and the actively transmitting number of STAs.

6.1 Header compression

6.1.1 Length reduction

Approximately one million frames were captured on heavy loaded networks with different types of applications. This was done on a public network (hot spot) as well as an office network AP. The frame headers were analyzed with a Packet Analyzer. All frames captured were sorted according to AC queue, source and destination and saved in separate files. The difference between two consecutive frames on the same file were calculated, byte by byte.

The captured frames were also compressed with the algorithms defined in section 4, using Perl scripts. The processing time of the Perl scripts was measured to be able to perform a relative comparison of the algorithms processing time.

I choose the algorithm parameters that provide the most header reduction for each case. I add the WLAN Header

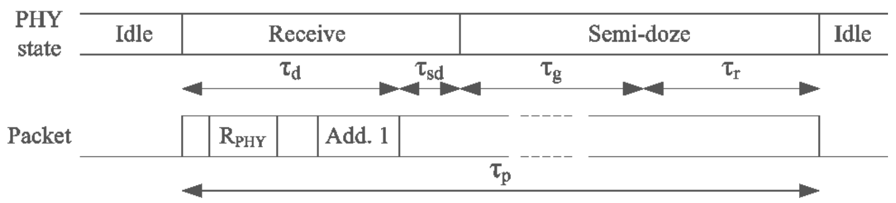

Figure 3: PHY states in a packet

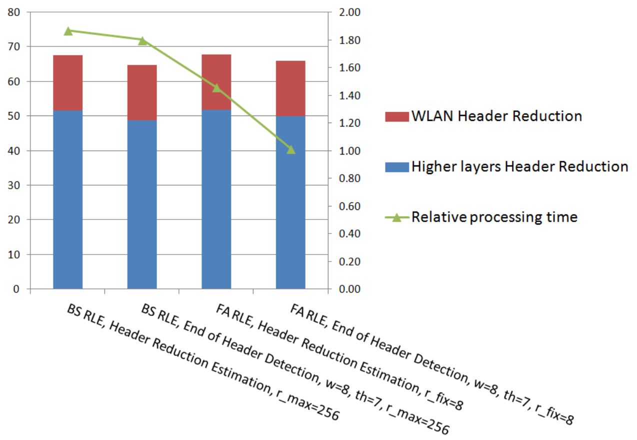

Figure 4: Overall best achieved Absolute Efficiency and processing time

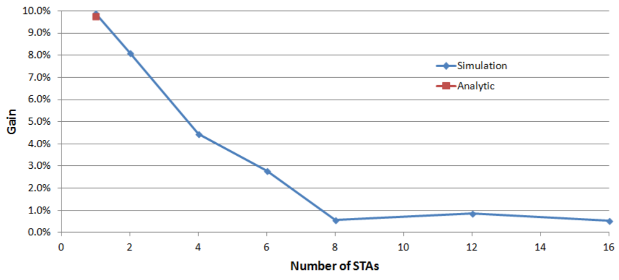

Figure 5: File Transfer with Header compression: power saved vs. number of STAs

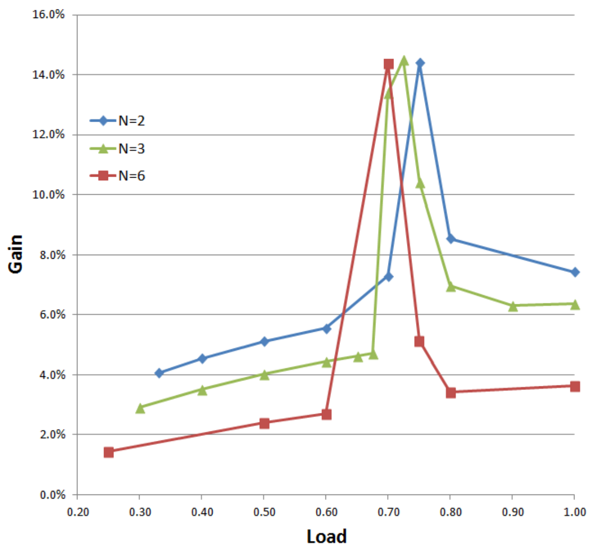

Figure 6: File Transfer with Header compression: power saved vs. load

reduction, resulting in the total achieved header reduction. The results are shown on Figure 4, where I depict also the relative processing time (excluding WLAN header compression, as it is identical for all cases). The Header Reduction algorithm detects the end of the header in a more efficient way than the End of Header Detection algorithm. I can see that there are only minor differences between the achieved header reduction; less than 3 bytes. But there are major differences in processing time: it takes almost twice longer to process the BS RLE with Header Reduction Estimation than the newly proposed FA RLE with End of Header Detection.

6.1.2 Power save

The simulations as well as the analytical model are in saturation mode: all STAs always have a new frame to transmit. I see that in this case the reduction in power consumption is negligible. The header compression only achieves to squeeze more payload per unit of time, which does not reduce the average active time. In real life, the user would like to transfer a file, with a finite size. I therefore will consider for the rest of this section the case where all STAs have to transfer a file of finite length.

I firstly vary the number of STAs that transfer their file simultaneously. The result is depicted on Figure 5. The simulation shows that the gain from using compression decreases when the number of active STAs increases. The number of collisions increase with the number of STAs. As the compression is done only on the first transmission attempt, the proportion of compressed frames decreases accordingly, bringing the gain to an asymptotic value of 0, for a big number of STAs. The analytical result was obtained using the simple single STA analytical model. The multiple STAs analytical model cannot be used as it assumes non-stop transfer instead of a finite length file transfer.

The next effect analyzed is the STAs load. It is possible to imagine, that the STAs do not have the file to transfer ready. The file needs to be preprocessed in chunks, before it is sent. Therefore, the data load seen by the transmitter may be lower. The results are depicted on Figure 6.

At low loads, the state of the STA is most of the time in idle. As the traffic is low, the impact of header compression on saved power is low. With the increasing traffic, the proportion of time in idle diminishes and so the gain achieved increases. At very high loads, the network is in saturation. There is no difference between a load that is slightly above saturation and 100% load; the same throughput is achieved and consequently the header compression power saved stays constant. There are many collisions, but thanks to the backoff counter range increase

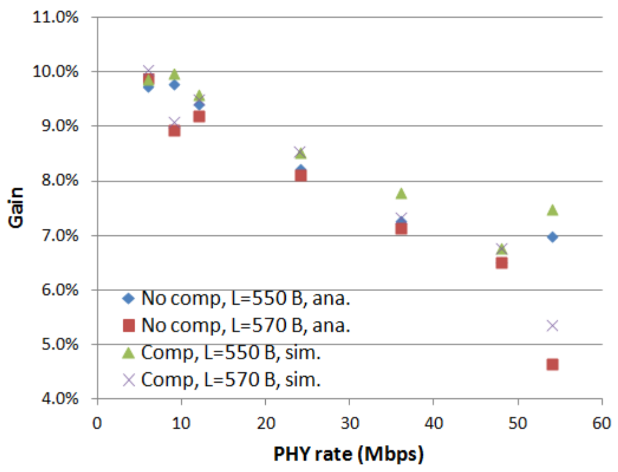

Figure 7: File Transfer with Header compression: power saved vs. PHY rate

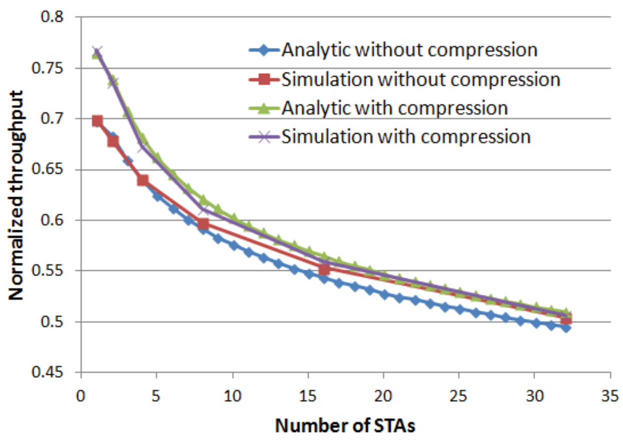

Figure 8: Throughput gain using header compression vs. number of transmitting STAs

(of the other STAs), still some frames are successfully transmitted at the first attempt, providing power saving. In order to understand the peak gain around a load of 0.7, it is necessary to check the throughput gain obtained by header compression on Figure 8. On this figure, it can be seen that if header compression is enabled, the maximum throughput is slightly greater. Therefore, for degenerate distribution (constant interval) packet generation and $N = 3$ STAs (for example), frames can be transferred without collision for a load up to 0.74 with header compression, while collisions start at a load of 0.68 without header compression. Therefore, for a load between 0.68 and 0.74, header compression saves power thanks to two reasons: shorter frames and less collisions. The second reason has a very high impact because frame retransmission consumes a lot of power.

The effect of the PHY rate is depicted in Figure 7. I can see a general behavior of reduced gain with increasing PHY rate. The cause is the OFDM symbol. For each PHY rate, a given number of bytes can be inserted per OFDM symbol. For example, only 3 data bytes are inserted per OFDM symbol, at 6Mbps, while 27 data bytes are inserted per OFDM symbol. Following the 802.11 standard, no partial OFDM symbol, may be transmitted. If the quantity of data bytes in a frame is not an integer number of OFDM symbols, then the last symbol is padded with zero bytes. For higher PHY rates, the average number of padded bytes becomes significant, compared to the header reduction achieved. Part of the bytes gained by header compression are reinserted due to the padding of the last symbol.

For example, at 54 Mbps, a frame of 650 bytes, requires 25 symbols (omitting the PHY header), with the last symbol containing only 2 data bytes and 25 pad bytes. The header compression reduces the header by 70 bytes. This brings the frame to 580 bytes, which require 22 symbols, with the last symbol half full. The achieved gain is 3 symbols. But if the frame length is slightly increased to 670 bytes, then the gain is only 2 symbols, and the last symbol after compression contains 21 padded zero bytes.

The same OFDM symbol padding causes the spread in gain between payload of 550 and 570 bytes, in Figure 7. The spread is wider at higher PHY rates, as the variation between no padding for the last symbol and almost complete padding reaches 26 bytes. This means that for some frame lengths, the header compression will be net 70 bytes, while in other extreme cases, it will only be 44 bytes, at 54Mbps.

6.1.3 Throughput gain

The first impact analyzed is the number of STAs. The results are depicted in Figure 8. I see a sharp decrease in throughput gain as the number of STAs increases. This is due to the increasing probability of collision and the fact that re-transmitted frames do not have their header compressed. When a single STA transmits, the normalized throughput is increased by 0.067 (representing almost 10%).

The effect of the frame length is depicted in Figure 9 for two active STAs. The general behavior is an increase of the throughput for longer frames, as the proportion of overhead to payload decreases. The throughput is increased by 0.075 for short frames (representing 24%), and slowly decreases to 0.035, for frames with 1400 Bytes of payload. This shows, that even for long frames, the header reduction creates a nice maximum throughput increase.

The effect of the header reduction is similar to the effect of the payload length, as it consists in varying the overall frame length. But the impact on the throughput is reduced because the variation on the frame length is limited.

The throughput gain is reduced with increasing PHY rate. The cause is similar to the one described in Section 6.1.2, for the power save gain: the OFDM symbol padding.

6.2 Packet overhearing reduction

This section compares the average power consumption of a standard STA and a STA that implements the packet overhearing reduction method.

Figure 10 shows the power consumption gain when using the packet Overhearing Reduction method in saturation mode with a PHY rate of 6Mbps. The power consumption decreases as the number of STAs increases. This can be explained by the larger average backoff counter value that

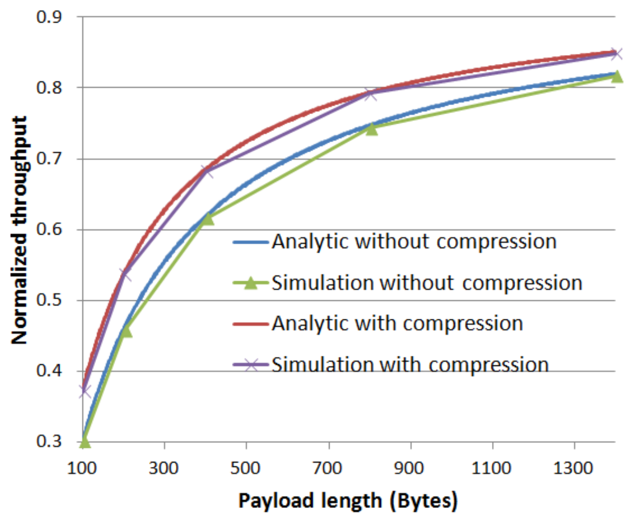

Figure 9: Throughput gain using header compression vs. payload length

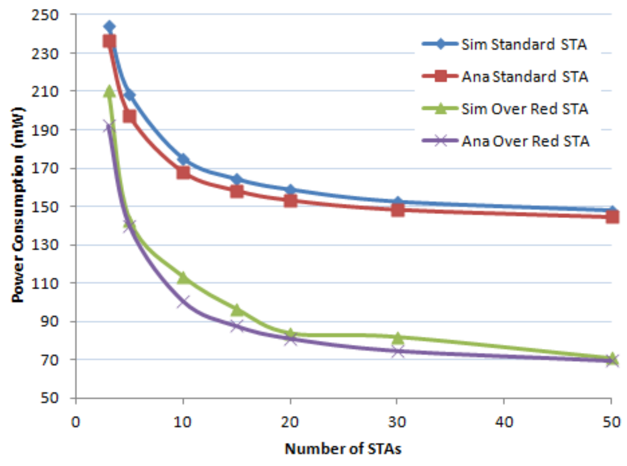

Figure 10: Power vs. N in saturation mode at PHY rate 6Mbps

the STAs use before they are allowed to transmit. Therefore, the STAs spend less time in transmit state (the most power consuming state) and more time in idle state. The amount of retransmissions, per STA, due to collisions in a channel with more STAs is higher. This gives more opportunities for a STA implementing Overhearing Reduction to be in semi-doze state. The power saved at lower PHY rate is larger thanks to the overall packet duration. At 6Mbps, the packets are longer, allowing the STA using Overhearing Reduction to stay longer in semi-doze state.

In Figure 11, the normalized throughput is fixed to 0.33, which is below the saturation throughput for the range of up to $N = 50$ STAs. For N small, the channel is far from saturation, therefore, the proportion of time a STA transmits is low. The standard STA spends therefore less time in receive state, which means also less time in semi-doze state for the STA implementing the Overhearing Reduction method. The result is less power save gain compared to the saturation mode for small N. As N increases, the channel approaches saturation and therefore, the power save gain approaches also the gain in saturation mode.

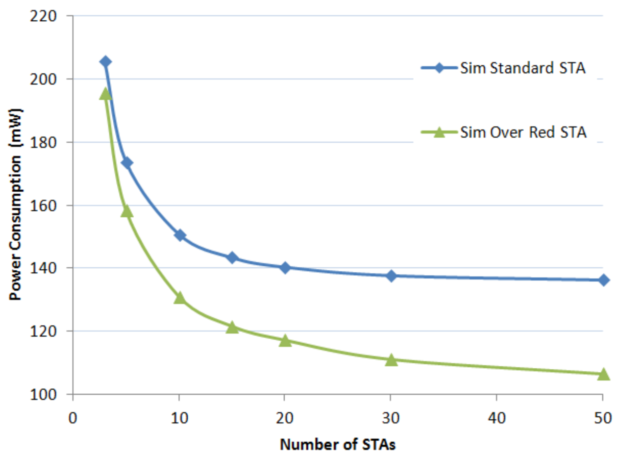

Figure 11: Power vs. N in non-saturation mode at PHY rate 24Mbps

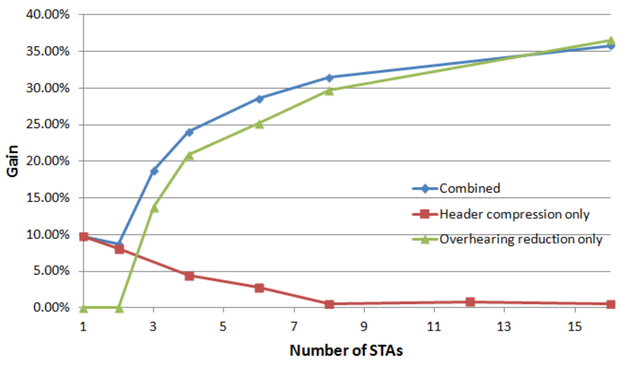

Figure 12: Combined methods, gain vs. number of STAs

6.3 Energy saved when combining both methods

In this section, I combine the methods of Header compression and packet overhearing Reduction. Packet overhearing reduction, by definition, cannot contribute to throughput increase. Therefore, I focus on power consumption analysis. As seen in Section 6.1.3, Header compression energy saving can be evaluated on finite size file transfer only. Therefore, this section will only analyze the energy saving on a file transfer.

The first impact analyzed is the number of STAs. The results are depicted in Figure 12. As seen in Figure 5, the energy saved from Header Compression decreases with the number of STAs actively transmitting. As seen in Fig 10, the gain increases with the number of STA. For $N = 1$ and $N = 2$, there is no overhearing, therefore, Overhearing Reduction is not possible. I can see that the methods complement each other. Where Overhearing Reduction cannot help, Header Compression provides its best performance. Where Header Compression cannot provide much energy saving, Overhearing Reduction takes over.

Based on the findings of Figure 12, when there are only one or two STAs active, only Header Compression brings energy saving. When there are many STAs active ($N > 8$), mainly Overhearing Reduction brings energy saving. For these cases, the advantages for Header Compression alone or Overhearing Reduction alone, are described in

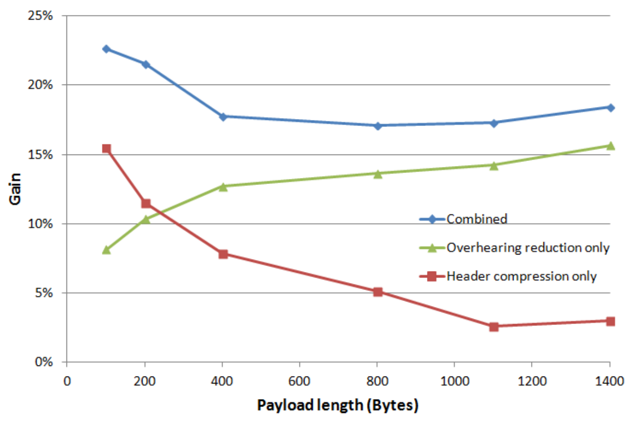

Figure 13: Combined methods, gain vs. payload length

Section 6.1.2 and Section 6.2 respectively. In this section, I will concentrate on the case where both methods contribute to the energy saving. I will, therefore, fix $N = 3$, to have approximately the same contribution from both methods, for the next simulations in this section.

The impact of the frame length is depicted in Figure 13. For Header Compression, the gain in energy decreases with the increased Payload length, as the proportion of the header to the payload decreases. For the Packet Overhearing Reduction, the longer the frame, the more energy can be saved, as the STA stays longer in semi-doze state, and less in receive state. I can see that also for Payload length, the methods complement each other, allowing for an almost constant gain over the complete Payload range.

Both methods have their gain reduced with reduced load. The reduced load translates to proportionally more time in PHY listen state compared to active states. Therefore, the gain achieved by header compression or packet overhearing reduction is negligible because it mainly reduces the time in PHY active states (transmit or receive).

7. Conclusion

The WLAN is a very popular way to transfer data from and to mobile devices at high rate without wires. In order to allow the user maximum flexibility, the interval between consecutive battery recharges should be increased as much as possible. This goal is achieved by reducing the energy consumption during data transfer. The goal of this paper was to define two methods to save energy during high data traffic. Namely, packet overhearing avoidance and all-layers header compression.

In order to estimate the energy saved by the proposed method, I developed an analytical model of a standard WLAN STA energy consumption. This model is derived from Bianchi's throughput model [2]. The analytical model was validated against an internally developed CSMA/CA simulator (in Matlab) and Bianchi's throughput model. I also modified this standard STA analytical model to support the header compression method and the packet overhearing reduction method.

I identified the cases to keep the probability low to have undetected errors in the headers. I defined an efficient compression of each WLAN MAC header field. I used the difference between consecutive frame of the same AC to compress unknown headers. The BS RLE algorithm was clearly defined and I introduced a new algorithm called the FA RLE algorithm. The new notion of boundary between unknown header and payload was introduced and three algorithms to identify this boundary were given. An analytical model was developed for the case of unidirectional communication between the AP and a single STA.

A method was described to reduce packet overhearing without the need of RTS/CTS frames. For this purpose, a new PHY state was introduced which is a compromise between the too slow doze state and the too much energy consuming idle state. I developed an analytical model considering the case where only the AP transmits frames to two STAs, meaning that no collision occurs. This model showed the asymptotic behavior of the energy gain.

The gains achieved by the two proposed methods was proven. A big amount of frames was captured on which the header compression algorithms were run. This gave the average expected frame size reduction that can be realistically achieved. I then used both the analytical model and the simulator to estimate the throughput gain and energy saved by reducing the frame size, thanks to header compression. The same tools were used to estimate the energy saved thanks to packet overhearing reduction. By combining both methods it could be seen how they complement each other. For small number of STAs transmitting, the header compression contributes more, while for longer frames, the packet overhearing contributes more. It is remarkable that at high load and low PHY rate, over a wide range of the critical remaining parameters, the achieved gain is above 15% if both methods are combined.

Acknowledgments. This work has been supported by Slovak National Research Grant Agency under project no. VEGA 1/0676/12. It is a partial result of the Research and Development Operational Program for the projects Support of Center of Excellence for Smart Technologies, Systems and Services, ITMS 26240120005 and for the projects Support of Center of Excellence for Smart Technologies, Systems and Services II, ITMS 26240120029, co-funded by ERDF.

References

[1] A. Amin, H. Qureshi, M. Junaid, M. Habib, and W. Anjum. Modified run length encoding scheme with introduction of bit stuffing for efficient data compression. *International Conference on Internet Technology and Secured Transactions*, 1:668–672, December 2011.

[2] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE J. Sel. Area Comm.*, 18(3):535–547, 2000.

[3] S. Biswas and S. Datta. Reducing overhearing energy in 802.11 networks by low-power interface idling. *IEEE International Conference on Performance, Computing and Communications*, pages 695–700, 2004.

[4] Bormann. *RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed*. Network Working Group, 2001.

[5] S. Casner and V. Jacobson. *Compressing IP/UDP/RTP Headers for Low-Speed Serial Links*. Network Working Group, 1999.

[6] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the I.R.E.*, page 1098–1102, September 1952.

[7] V. Jacobson. *Compressing TCP/IP Headers for Low-Speed Serial Links*. Network Working Group, 1990.

[8] S. Kodituwakku and U. Amarasinghe. Comparison of lossless data compression algorithms for text data. *Indian Journal of Computer Science and Engineering*, 1(4):416–425, 2010.

[9] B. McFarland. How to use optional wireless power-save protocols to dramatically reduce power consumption. *EETimes magazine*, May 2008.

[10] R. Palit, K. Naik, and A. Singh. Impact of packet aggregation on energy consumption in smartphones. *International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 589–594, July 2011.

[11] S. Tang, H. Yomo, Y. Kondo, and S. Obana. Wake-up receiver for radio-on-demand wireless lans. *EURASIP Journal on Wireless Communications and Networking*, February 2012.

[12] T. A. Welch. A technique for high-performance data compression. *IEEE Computer*, 17(6):8–19, June 1984.

[13] Z. Zeng, Y. Gao, and P. Kumar. Sofa: A sleep-optimal fair-attention scheduler for the power-saving mode of wlans. *International Conference on Distributed Computing Systems*, pages 87–98, June 2011.

Selected Papers by the Author

D. Levy, I. Kotuliak. WLAN power consumption analytical model. In *Wireless and Mobile Networking Conference*, pages 101–106, Bratislava, Slovakia, 2012.

D. Levy, I. Kotuliak. General all-Layers combined with Efficient WLAN MAC Layer Headers Compression. In *International Conference on Advances in Mobile Computing and Multimedia*, LNCS 5164, pages 123–132, Vienna, Austria, 2013.

D. Levy, I. Kotuliak. WLAN Power Saving Using Packet Overhearing Reduction. In *Telecommunication Systems Journal*, accepted for publication in Springer.

D. Levy. DSI3 Sensor to Master Current Threshold Adaptation for Pattern Recognition. In *Int. Conf. on Signal Processing Systems*, Sydney, Australia, 2013, obtained the "Excellent Paper Award". Published in *Int. Journal of Signal Processing Systems*, pages 141–145. Engineering and Technology Publishing, 2013.

D. Levy. DSI3 Sensor to Master Decoder using Symbol Pattern Recognition. In Society of Automobile Engineers, editors, *Int. J. Passeng. Cars – Electron. Electr. Syst.*. SAE, Warrendale (PA, USA), 2014.