

Designing Energy Efficient Automatic Repeat Request Protocol in Wireless Sensor Networks

Cheng Guo
Delft University of Technology
Mekelweg 4, 2600 GA, Delft,
The Netherlands
c.guo@tudelft.nl

Przemysław Pawełczak
Delft University of Technology
Mekelweg 4, 2600 GA, Delft,
The Netherlands
p.pawelczak@tudelft.nl

R. Venkatesha Prasad
Delft University of Technology
Mekelweg 4, 2600 GA, Delft,
The Netherlands
r.r.venkateshaprasad@tudelft.nl

Ramin Hekmat
Delft University of Technology
Mekelweg 4, 2600 GA, Delft,
The Netherlands
r.hekmat@tudelft.nl

ABSTRACT

Energy constrained Wireless Sensor Networks (WSNs) challenge traditional protocols in many aspects. One such challenge is the requirement of energy efficiency in the Automatic Repeat Request (ARQ) protocols. Although traditional ARQ protocols are optimized for a better throughput, they fail to minimize the energy consumption. In some applications in sensor networks, delay can be tolerated. Using this aspect, we propose a link quality aware ARQ (LQ-ARQ) protocol which only retransmits packets from a buffer when the link from the sender to the receiver is relatively good. The link quality is judged by checking the Received Signal Strength (RSS) of the data packets, thus no extra overhead is introduced. To avoid using an inflexible threshold of RSS to judge a good or bad link, which is hard to decide in practice, we propose a machine learning mechanism to decide the threshold softly depending on link quality. Our protocol is able to deliver packets reliably with least amount of retransmissions. Thus energy consumption is reduced. Moreover, usage of limited resources such as memory is minimized in the protocol. We compared our protocol with traditional ARQ protocols with real field experimental data. Results show significant improvements.

Categories and Subject Descriptors

C.2.2 [Computer-Communication Networks]: Network Protocols

General Terms

Algorithms, Experimentation, Performance

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHANTS'09, September 25, 2009, Beijing, China.

Copyright 2009 ACM 978-1-60558-741-7/09/09 ...\$10.00.

Keywords

Wireless Sensor Network, ARQ, Energy Efficiency, Link Quality, RSSI

1. INTRODUCTION

Wireless sensor networks normally have limited memory, energy and computational power. In a network of sensors, with the objective of transmitting the sensed data to the sink, the packets are transmitted multiple times when an Acknowledgement (ACK) is not received. When the packet loss is random and isolated multiple transmissions help. However, when the link is in deep shadow transmitting multiple times can not deliver the packet. Therefore without considering the link state, the traditional Automatic Repeat Request (ARQ) protocols, which retransmit a lost packet multiple times, are deficient in terms of energy efficiency. For example as in 802.11 protocol a lost packet is retransmitted eleven times before the sender gives up, thus consuming too much of energy. Meanwhile, a number of WSN applications do not require real time transmission. For instance, a WSN tracking the air pollution index in a built up area is not substantially concerned if the carbon dioxide index of the moment is delayed by a minute or two before sending it to the sink node. Such applications sense the environment periodically and the sensed data can tolerate a certain degree of delay in transmission. Another potential example can be the delay-tolerant networks. In such networks a node does not have fixed links to its "neighbors" because of movement of nodes in the network or due to the unavailability of the links. Thus their neighbors may show up or disappear from time to time. Therefore, in these scenarios, traditional ARQ protocols fail to deliver a packet while wasting a lot of energy on retransmissions.

Delivering packets efficiently and reliably is the primary concern. It is futile to attempt to transmit the packets when the link is not good. Thus we try to explore the idea of estimating the packet delivery ratio and attempt to transmit packets only if the chances of success are higher. In fact one can observe that there is a trade-off between saving energy and the latency in transmission. Depending on the high level goals one can tune the network to deliver the data faster or live longer. It is particularly beneficial to have tools

Table 1: Major commercially available WSN platforms and their parameters

Platforms	Processor	RAM	Programming ROM
TMote	8MHz 16bit	10kB	48kB
Mica2	16MHz 8bit	4kB	128kB
IRIS	16MHz 8bit	8kB	128kB
BTNode	8MHz 8bit	4kB	128kB

or parameters under control to achieve the requirements in various expected scenarios. In this paper we define a simple protocol in which the packets are stored in a buffer – when undelivered – until it sees a good enough channel. We note that our protocol is suggested to be implemented in MAC layer and only deals with the reliable transmission on a link. However, the ideas are general and it could be implemented in higher layers too.

We also note here that the energy is not the only limited resource in sensor networks, computational power and memory also need to be taken into account. For example, we list the following four major commercial WSN platforms, i.e. Tmote [1], Mica2 [5], IRIS [4] and BTNode [2] in Table 1. From the table we can see that most of the platforms have micro processor of limited computational ability, thus the protocol designed should have low complexity. Moreover, the RAM and ROM are so small that can be easily filled up. For instance, if a packet is 50 Byte and we have the buffer for 40 packets, then we would have already used 2kByte RAM. Therefore the designing of an energy efficient protocol should not only focus on the performance, such as delivery ratio and energy consumption, but also the feasibility of such a protocol. It has to have a simple decision making algorithm for retransmitting packet and the buffers have to be of limited size.

An energy efficient ARQ protocol should only retransmit a lost packet when the channel is good enough. Otherwise it is a waste of power at both the transmission and reception ends. The protocol should work well in realtime in the field. In other words, the performance of the protocol should be checked with multiple real life scenarios instead of using only a certain wireless channel models. In populated area where WSNs are normally deployed, wireless link varies a lot with time.

While designing the protocol, we have to decide whether a link is good or not. Thus a threshold has to be set. However, if we try to fix it with experimentally found values, it does not guarantee that the protocol would work in all the scenarios. This is due to the complete dynamic characteristics of these links. Since the protocol has to work in various scenarios, it should be free of this kind of parameters or the parameters should be decided on the fly. All the requirements above, energy efficiency, reliable delivery, computational and memory constraints and variation in the environment, bring us challenges in designing the protocol.

After reviewing briefly the related work in the literature in Section 2, we introduce the design principles of the proposed protocol, Link Quality aware ARQ (LQ-ARQ), in Section 3. In Section 4 we show the performance of the proposed protocol with the experimental data and compare it with other ARQ protocols. We conclude the paper in Section 5.

2. RELATED WORK

Authors in [13, 14] have looked into the tradeoff between energy consumption and throughput. They modeled the throughput and energy consumption using a Markov model. Then they proposed an easy probing based approach such that whenever an ACK is missed, the transmitting node switches to a probing mode in which only pilot packets are sent periodically until an ACK is received. The idea is to reduce the energy consumption by probing link quality with small pilot packets. However, this protocol assumes that: 1) the channel can be modelled precisely with the markov chain; 2) the buffer size is unbounded; and 3) the transmitter has a lot of packets to send thus the pilot packets are sent with a much lower rate than the data packets. In a real WSN, say, monitoring environmental parameters, these assumptions may not be true. In fact most of the WSNs are not designed for the heavy traffic. The aim is to conserve their energy and transmit the important changes in the environment.

Inspired by the above work, the authors in [8] proposed channel aware enhancements to the Stop&Wait ARQ. Based on the assumption of Rayleigh fading, instead of sending the probing packet periodically, they adjust this period based on the length of the fading, which is decided by considering a Rayleigh fading model. The authors have proposed that the waiting time till the next probing is decided by the state of the current fading, i.e. the start of the fading, the middle or the end. They showed some improvements with their model. However, the authors did not check whether the model represents the real scenarios all the times. The pilot packets and the data packets would also be of the same size in WSNs thus it may not decrease the bandwidth. Moreover, detecting the fading state is non-trivial and an expensive task for WSNs from the point of views of the computation and power. The authors in [10] proposed a machine learning algorithm to decide whether to retransmit. A decision regarding retransmission is awarded or punished depending on whether an ACK is received. Their decision making algorithm is actually a variation of using Exponential Weighted Moving Average (EWMA) filtering to estimate the Packet Delivery Ratio (PDR). A successful delivery will rise the estimation of the current PDR with a fixed amount while a failure will decrease the estimation by the same amount. The impact of previous estimation decrease exponentially with time. The difference is that the authors add several more parameters into the algorithm which makes the rate of change in increment and decrement different. However, the parameters used in adaptation are hard to decide, a pre-determined value only fits the already available measurement of certain environment while deciding the values of these parameters on the fly—that models the current scenario properly – is highly difficult to implement.

Many of the previous studies found in the literature are based on an assumed model [8, 10, 13, 14]. Depending on how the model depicts the reality in practice, the performance improvement of these protocols varies in real environments. We have found in our extensive measurements that no two experiments give the similar measurements. The variation in the signal strength and PDR are high and thus fitting a predetermined model will not work in most of the cases. Therefore, our objective here has been on two facets: (1) use the current situation of the environment effectively and take the decision based on the current measurements; (2)

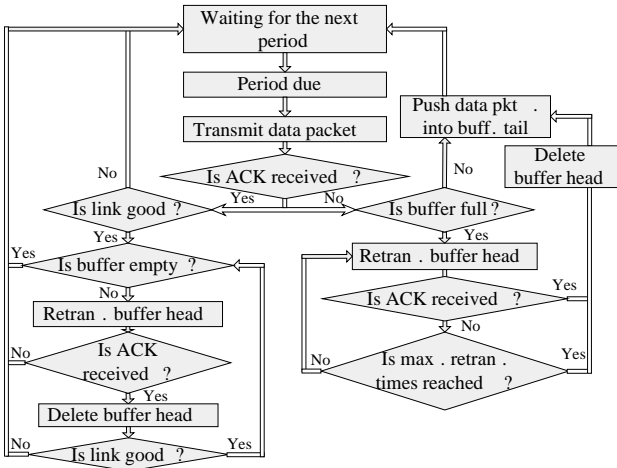


Figure 1: Flow chart of LQ-ARQ

use the fact that WSNs usually tolerate certain amount of delay so that buffering could be used to exploit the link to fullest when it is most useful. Further, our objective is also to design a low complexity protocol which uses small amount of memory. We try to deliver a packet with least number of retransmissions. Meanwhile, the protocol should also minimize the probability of dropping packets.

3. LINK QUALITY AWARE ARQ PROTOCOL (LQ-ARQ)

The basic idea of the proposed protocol is to retransmit failed packets only when the link is good. Thanks to the small packet size in WSNs, we use data packet to probe a link instead of dedicated pilot packets. It allows us to reduce the energy consumption of transmitting the pilot packets. A sender in the proposed ARQ protocol by default operates in the Normal Mode (NM) and transmits the sensed data periodically. If an ACK is not received, it stores the failed packets in its buffer. Otherwise, from the received ACKs, the sender may be informed about the link quality which the receiver sees through the measure of signal strength. It is to be noted here that the link quality is found at the receiver as the receiver sees at the instant. Now, if the link is bad, it goes back to sleep and waits till the next period. If the link is good, the sender enters a mode which we denote as Retransmission Mode (RM) and retransmits the buffered packets. The sender quits from RM to be back to NM in two cases, the link is indicated as bad or the buffer is empty. If a good link can not be found before the buffer overflows, the sender retransmits the first buffered packet a certain number of times before dropping it. We drop the oldest packet in the buffer since its usefulness is limited. Fig. 1 illustrates the working procedures of our protocol. The performance of the protocol is decided by the feasibility and the efficiency of the following three factors: link quality estimation, buffer size and retransmission procedure.

3.1 Link Quality Estimation

Since the efficiency of the protocol depends on whether or not we can precisely find when the link is good. The link quality estimation plays an important role in our LQ-ARQ.

There are generally two sources that we can use to estimate PDR, namely data packets and Receiving Signal Strength Indicator (RSSI) of received packets. We introduced their strengths and weaknesses in [9]. If a link has an on-off characteristics, which means that we can either send packets with 100% PDR or 0%, a successfully acknowledged packet already indicates a good link. Thus if a packets is acknowledged, we consider the link as good otherwise as bad. This method is rather simple, however the accuracy may not be high since we may have intermediate links.

Another option is to use the RSSI of the received data packets to estimate the PDR. In [6, 11, 12] it is observed that there exists a strong correlation between PDR and RSSI when RSSI is high. We also confirm the same observation with our measurements. Fig. 2 shows the RSSI to PDR mapping in a mobile experiment, in which the receiver was fixed and the sender was moving to and away from the receiver. The mapping tells us what the PDR is in a slot in which the RSSI is collected. We can see that when the RSSI is higher than -80dBm, RSSI can be precisely mapped to the PDR, which is close to 1, with a small variation. Since we want to know when the channel is reasonably good for transmission, using the high RSSI values to estimate the PDR serves this purpose well.

We can read from the RSSI value if the attenuation of signal is more severe than a threshold. However, if we have a hard threshold as a criterion, say for example when RSSI is higher than -80dBm, the sender may not have a chance to send the buffered packets in case the transceiver pair was deployed far apart or a lot of multipath or interference is present thus the RSSI is never as high as -80 dBm. Another important aspect is that mere RSSI being low will not preclude a proper transmission at all the times. Sometimes packets do get across even if RSSI is low. So instead of a hard threshold, we use a soft threshold by looking into the past received RSSIs. If the just received RSSI is among the T percent of highest RSSIs in the past, then we consider the channel as relatively usable. It may be the case that even with a RSSI among the T percent of the highest the delivery ratio may not be enough. However, the past measurements tells us that all the time RSSI is not always as high as one can expect it to be. So the node should take the opportunity to transmit all the buffered packet.

Since we want the LQ-ARQ protocol to be such one that it should decide the soft threshold by itself in different scenarios. To do this, we implement a simple machine learning mechanism. If the threshold is set too high, then we merely find a good link so that packets will be accumulated in the buffer which may overflow frequently. Therefore we have to reduce the threshold by m to "find" more good links. Contrarily, if the threshold is set too low, the protocol may be misled to retransmit buffered packets when the link is not good thus the link is indicated as good in the RM mode. In this case we should increase the threshold by n so that the protocol is more careful to select a good link. Once on the verge of being buffer overflow, the packet in the head of the buffer has to be retransmitted, a fixed, k times before being dropped, thus we may waste k retransmissions. If the link is indicated as bad in RM mode, one retransmission is wasted. Therefore m should be k times of n . n should be small so that the threshold does not jump heavily. We will evaluate the machine learning mechanism in Section 4.

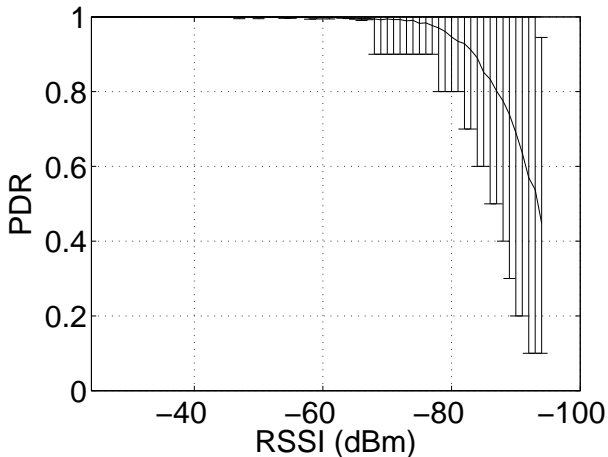


Figure 2: RSSI to PDR mapping, the solid line is the average and the bar indicates 90% percentile

3.2 Buffer Size

As introduced in Section 1, we need buffers to store the failed packets. Due to the limitation of the hardware, we can not assume that the buffer is infinite. In other words, we have to try to find the sufficient size of the buffer while keeping it as small as possible. So the question is to quantify the effect of this limited buffer on the performance of an energy efficient ARQ protocol. If the “good” state of a link appears frequently, then we do not need a large buffer. Contrarily, if the “bad” state continues for a long time, we have to have a big buffer.

In our ARQ protocol, the sufficient size of the buffer is actually decided by the effectiveness of the learning mechanism we introduced in the last subsection. If the threshold is set too high by the mechanism, even though the link is good in reality, the sender would be restrained to retransmit the buffered packet. Thus the buffer has to be large. However, if the threshold is set too low, although the buffer size can be reduced, the protocol may retransmit in bad state thus decrease the energy efficiency. The buffer size can be either decided by physical considerations such as available memory on the device, i.e. the maximum possible due to hardware platform or by measurements in different scenarios, which does not affect the limited memory while keeping the performance of the LQ-ARQ satisfactorily.

3.3 Retransmission Methods

Although the protocol should have found a good link, the retransmission may still fail due to the imperfect estimation of the link. Thus we have to select a retransmission method, i.e. stop and wait or selective repeat¹. In the first method retransmission of a packet is followed by waiting for the ACK before the next retransmission. The received packet is dequeued from the buffer. If the acknowledgement is not received, it quits from RM to NM. In the second method transmission of all the buffered packets are done back to back then the sender waits for an ACK, which indicates the failed packets by their IDs. Then the sender deletes the ac-

¹We assume packets are tagged with packet ID and it is not necessary to keep their sequence. Thus Go Back N is not considered here.

knowledge packets and keep the failed packets in the buffer. If the link has the bad state more often then we expect first method to be more efficient since it is more cautious in retransmitting. On the other hand, if the link has more good state, the second method may catch more opportunity to retransmit thus avoiding overflow. Again, we will check their performances and select one by measurement results.

4. MEASUREMENTS AND NUMERICAL RESULTS

Although the protocol is parameter-free, we are curious to find the sufficient size of the buffer and the better retransmission method. Moreover, we have to evaluate the LQ-ARQ protocol. We should certainly implement the protocol in an experimental testbed, then collect the experimental data to analyze the performance. However, in this case it is hard maintain the same environment for all the experiments for a long time. Thus the results may not be statistically comparable. Moreover, we may have to try different configurations to find sufficient buffer size, which is time consuming. Alternatively, we can run the experiments and record the PDRs and RSSIs in different scenarios then use the recorded data to run a simulation. This measurement data driven simulation enables us to repeat the experiment for the same environment.

4.1 Experiment Setup

To acquire the experimental data, we use t-mote sky sensor motes with 2.4 GHz IEEE 802.15.4 compliant Texas Instruments CC2420 transceivers [3]. The transceiver provides RSSI in the range of 100 dBm for every received packet, which is the received signal strength reading averaged over eight symbol periods. The IEEE 802.15.4 packet header is 17 bytes and the payload is set to 18 bytes thus a packet is 35 bytes long. We consider this length as typical for WSN packets. Only one packet length was used in the experiments since in [7] authors have shown that packet size does not affect PDR much in WSN unless otherwise it varies considerably.

We do the experiments in three scenarios, which we consider representing three challenging working environments in real life. They are introduced below.

1. *Weak link*: The experiment was performed in the corridor of TU Delft Wireless and Mobile Communications Lab, which was about 100 m long and 2 m wide, with offices at either side. One node was kept in a office and the other in the corridor, thus no LOS existed. The distance between nodes were deliberately tuned to be about 25 m so that the PDR varied a lot in the link. The packet collection lasted for about 2.5 hours.
2. *Mobile link*: Again, the same corridor was used as the experimental environment. The transmitter was placed on a table in an office and the receiver was carried with a walking speed of approximately 1 m/s. The office door were kept open in the experiment. We started from the fixed node and walked out the office with the mobile node to the corridor, where we walked at an even pace for 30 m and turned around to be back in the office. At the farthest point the PDR was expected to drop to zero.

Table 2: Simulation Configuration

Buffer Size	5, 10, 15, 20, 25, 30, 35, 40 45, 50, 60, 70, 80, 90, 100 200, 300, 400, 500, 1000
RSSI History Length h	120
Maximum Retran. k	3
Threshold Decrement m	0.03
Threshold Increment n	0.01
Period	1 s

3. *Dynamic link*: The sender and receiver were placed in the EEMCS faculty canteen of TU Delft. The packet collection started at 11:00 AM and lasted for 2.5 hours. There was LOS between the nodes, which were placed 20 m apart. However a lot of interruptions were expected during the lunch time due to frequent movement of staff and the students.

We emphasize that the first two experiments were conducted in a quiet period after working hours. Thus we expected no intensive interruption from the movement of people. Also, in all three experiments, we operated in a 2.4 GHz range absent from WLAN activity to minimize interference from other wireless systems.

4.2 Simulation Results

To evaluate the performance of LQ-ARQ, we define Retransmission Ratio (RR)

$$RR = \frac{\text{Retransmission number}}{\text{Lost packet number}}.$$

If a sender transmits 1000 packets, 200 of them are not received and the sender retransmits 400 times to deliver these packet then the RR is 2. The most energy efficient protocol should have a RR of 1. The configuration are listed in Table 2. We set the RSSI history length of 120 entries. Even we receive a packet every second, we have the RSSIs received in the last 2 minutes, which we consider sufficient to judge the current RSSI but not occupy too much RAM, i.e. 120 bytes. We set the maximum retransmission as k , which is same as the one suggested in 802.15.4 standard. The threshold increment n , as introduced in Subsection 3.1 is 0.01 which changes the threshold smoothly. In fact, as long as we keep the increment small enough, say, among 0.01 and 0.03, the performance of LQ-ARQ does not change much. The decrement m is k times the increment, 0.03. The sender transmits one packet per second to the receiver thus the period is one second. This packet is randomly selected from the 10 packet in the second which are collected from the experiments. If this packet is lost, we set the RSSI as -101dBm to mark the packet as lost. Otherwise, the actual RSSI is used to judge the link quality.

Firstly we would like to find the sufficient buffer size and the retransmission method. We can imagine that the larger the buffer the smaller is the RR. Fig. 3 shows how the RR changes in the three scenario when we increase the buffer size from 5 to 1000. Please note that the increment of the buffer size on X-axis is not linear. We can see that both the slopes of the solid curves of the Stop&Wait method in dynamic and mobile links turn from high to low at 35. Assuming each packet is 50 bytes on the average, it is an affordable

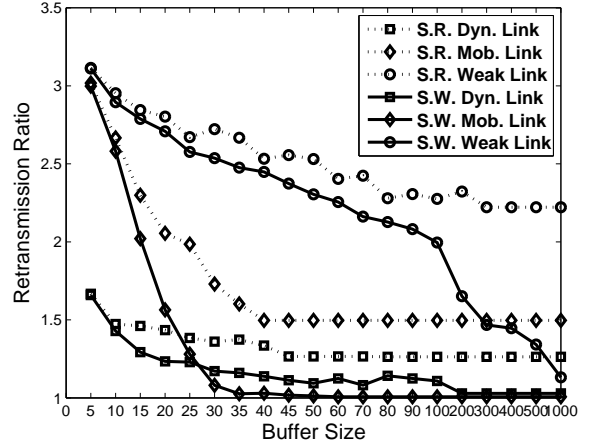


Figure 3: Optimum buffer size with respect to the retransmission ratio

buffer size for most of the platforms listed in Table 1. However, in the weak link, we have to set the buffer size to 500 to get a low retransmission ratio. It is not acceptable for a sensor platform. Therefore we set the buffer size 35 for all the three scenarios in the rest of the paper while presenting our results unless stated otherwise. The dash curves shows the RR of the Selective-Repeat method. In all the three scenarios and all the buffer sizes, the Selective-Repeat method performs worse than the Stop&Wait method. The reason is that when the link quality is bad, the link state indicated as good may not be actually good since they are selected with respect to the RSSI in the history. In this case, a Stop&Wait method stops retransmitting when an ACK is missed. However, Selective-Repeat does not do so. Therefore many buffered packets retransmitted at that time slot are lost thus the RR increases.

Secondly we want to check the effectiveness of the machine learning mechanism we have introduced into the system. It should be able to tune the threshold to the optimum one. We change the threshold in the three scenarios from 0 to 1 to find the lowest RR respectively then compare to the RR found by the learning mechanism. Fig. 4 shows how the RR fluctuates with different thresholds. The horizontal lines in the figure show the RRs from the machine learning mechanism. We can see that each line cut the lowest point of the experimentally found stable threshold curve respectively. Thus the mechanism can tune the threshold to the optimum one. Fig. 5 shows how the threshold decided by the machine learning mechanism changes with time in the mobile link. We set the initial threshold as 0.5. By the mechanism, the threshold quickly decreases then fluctuates around the optimum threshold, 0.35, as shown in Fig. 4. Therefore, the machine leaning is proven to be effective for easy deployment. The learning algorithm is also lightweight.

Next we evaluate the performance of the LQ-ARQ protocol. We choose two methods of link quality assessment introduced in Section 3, namely data packets and RSSI. In the first method, we consider the link is good if we have a packet acknowledged. The second method uses RSSI as the source and implements the machine learning mechanism. We use the normal Stop&Wait ARQ as the benchmark to compare

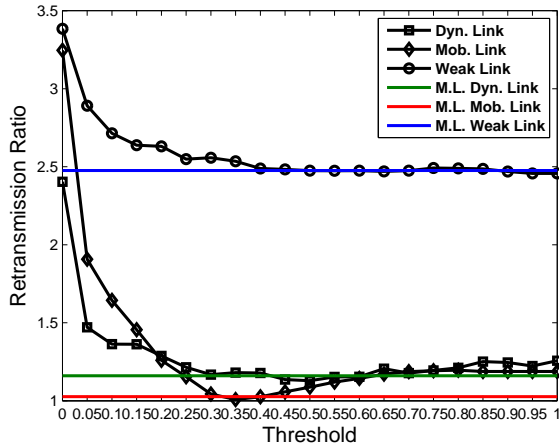


Figure 4: Optimum threshold with respect to the retransmission ratio

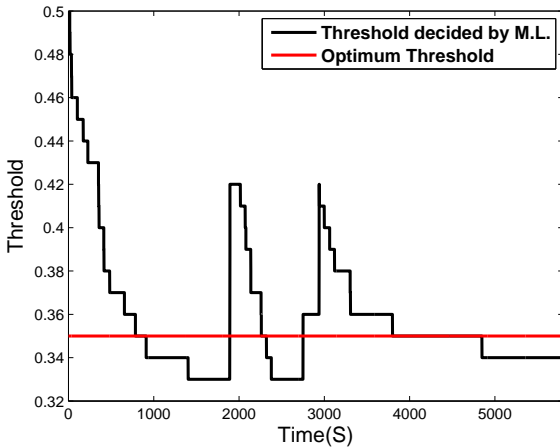


Figure 5: Threshold decided by the machine learning mechanism

with the two variants, namely data and RSSI, of LQ-ARQ. We set the buffer size of the LQ-ARQ as 35. Fig. 6 shows the RR comparison of Stop&Wait protocol and the two variants of the LQ-AQR. We can see that in each scenario, the traditional Stop&Wait protocol fails to achieve energy efficiency. It retransmits many times on an average since it does not judge link quality when it retransmits. Both the two variants of LQ-ARQ manage to achieve a very low RR in dynamic and mobile link. However, in the weak link, the performance is much worse. The reason is that the link quality in the weak link was too bad. We can see in Fig. 4 that the optimum threshold in this scenario is 1. It means that whenever a packet is acknowledged, no matter what the RSSI is, the protocol should consider that the link is good and take the chance to retransmit the buffered packets. The RSSI variant of LQ-ARQ turns to the data variant in this case. It also explains that the RR of the two variants in this scenario are the same. As shown in Fig. 3, the only way to reduce the RR is to increase the buffer size which allows lost packets stay longer in the buffer waiting for a good link state.

Besides the RR, we are also interested in the number of dropped packets of LQ-ARQ, which is the packets dropped from a buffer after trying to delivered the packet for maximum k times. We also show the number of packets which is lost in the first transmission which is indicated by the “Transmit data packet” block in Fig. 1. We denote them as lost packets. One can imagine that a low number of dropped packets leads to a low RR because each packets is retransmitted three times before dropping in our simulation. Fig. 7 shows the number of lost and dropped packets in the three scenarios. Again, the number of dropped packets of LQ-ARQ is very low, only a few compared to the thousands of lost packets in the traditional Stop&Wait protocol. Since it does not have a buffer to let lost packets wait for a good link state. In the weak link, although the number of LQ-ARQ is lower compared to Stop&Wait, the protocol still drops quite significant amount of packets.

We prove with experimental data that both variants of LQ-ARQ are able to achieve very low RR and number of dropped packets. Thus the energy spent to transmit these redundant packets are saved and high energy efficiency is achieved. The complexity of the data variant is so low that it can be easily implemented in any sensor platform but performs very well. If a bit more complexity can be tolerated, the RSSI variant is even more efficient in RR.

5. CONCLUSIONS

In Wireless Sensor Networks (WSNs) energy efficiency is important since they have to live a long time with batteries. The ARQ protocols do not consider the energy efficiency in the design hitherto. We have presented a Link Quality aware ARQ (LQ-ARQ) method that takes energy consumption into consideration. It optimizes the power consumption by reducing the total number of retransmissions. The simple idea behind this ARQ protocol is to buffer the data for certain amount of time since in WSNs delay can be tolerated. We retransmit the buffered packets only when the link is good. The link quality is judged by checking the Received Signal Strength (RSS) of the data packet, thus no extra overhead is introduced. We have also designed an algorithm that can work without any hard threshold. We tested our design in real environment and compared the traditional

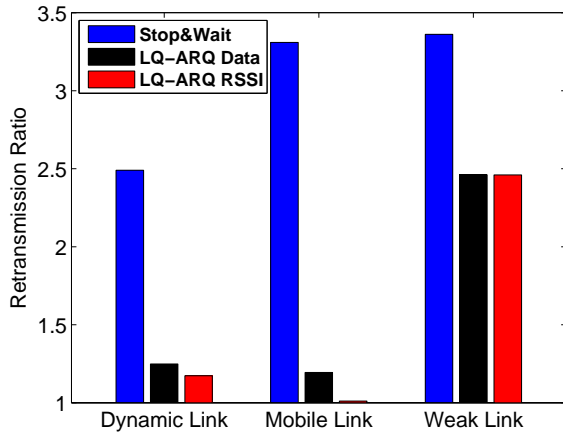


Figure 6: Retransmission ratio comparison in the three scenarios

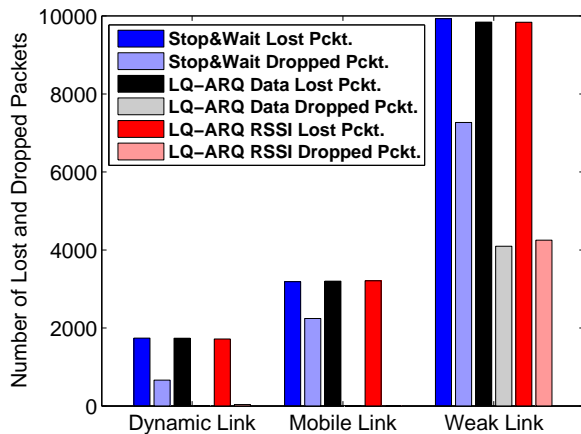


Figure 7: Number of lost and dropped packet comparison in the three scenarios

Stop&Wait protocol. We show that LQ-ARQ performs multiple times better than the Stop&Wait in the retransmission ratio. Moreover, we show that if the link is not very bad, LQ-ARQ merely drops any packet while the Stop&Wait drops thousands of packets in the similar situation. In future, we want to implement the protocol in a real WSN device and test its performance. Moreover, since LQ-ARQ introduces extra delay in MAC layer, the routing layer may retransmit without knowing the situation in MAC. Thus the MAC layer may be congested. We would like to propose a cross-layer solution to make the two layers cooperate together.

6. REFERENCES

- [1] Tmote sky datasheet, <http://www.moteiv.com/products/docs/tmote-sky-datasheet.pdf>, 2006.
- [2] Btnode rev3.24 product brief, http://www.btnode.ethz.ch/pub/uploads/main/btnode_rev3.24_productbrief.pdf, 2007.
- [3] CC2420 RF transceiver datasheet, <http://focus.ti.com/docs/prod/folders/print/cc2420.html>, 2008.
- [4] Iris datasheet, http://www.xbow.com/products/product_pdf_files/wireless_pdf/iris_datasheet.pdf, 2008.
- [5] MICA2 datasheet, http://www.xbow.com/products/product_pdf_files/wireless_pdf/mica2_datasheet.pdf, 2008.
- [6] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *Proc. ACM SIGCOMM'04*, Portland, OR, USA, Aug. 30 – Sep. 3, 2004.
- [7] A. Cerpa, J. L. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links: Modeling and implications on multi-hop routing. In *Proc. ACM MobiHoc'05*, Urbana-Champaign, IL, USA, May 25–28, 2005.
- [8] S. De and H. D. Cavdar. Channel-aware link layer arq strategies in wireless networks. In *Proc. IWCMC '08.*, Crete Island, Greece, Aug. 2008.
- [9] C. Guo, J. Zhou, P. Pawelczak, and R. Hekmat. Improving packet delivery ratio estimation for indoor ad hoc and wireless sensor networks. In *Proc. IEEE CCNC 2009*, Las Vegas, Nevada, U.S., Jan. 2009.
- [10] K. S. Kumar, R. Chandramouli, and K. P. Subbalakshmi. On stochastic learning in predictive wireless ARQ. *Wirel. Commun. Mob. Comput.*, 8(7):871–883, 2008.
- [11] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. Some implications of low power wireless to IP networking. In *Proc. HotNets-V*, Irvine, CA, USA, Nov. 29–30, 2006.
- [12] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proc. ACM SenSys'03*, Los Angeles, CA, USA, Nov. 5–7, 2003.
- [13] M. Zorzi and R. R. Rao. Energy constrained error control for wireless channels. 4:27–33, 1997.
- [14] M. Zorzi and R. R. Rao. Error control and energy consumption in communications for nomadic computing. 46:279–289, 1997.