An Energy Efficient Hybrid Interference-Resilient Frame Fragmentation for Wireless Sensor Networks

Ammar Meer*, Anas Daghistani*§, Basem Shihada*
*CEMSE Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia
§Department of Electrical and Computer Engineering, Purdue University, Indiana, USA
ammar.meer@kaust.edu.sa, anas@purdue.edu, basem.shihada@kaust.edu.sa

Abstract—**Frame fragmentation into small blocks with dedicated error detection codes per block can reduce the unnecessary retransmission of the correctly received blocks. However, the optimal block size varies based on the wireless channel conditions. Further, blocks within a single frame may have different optimal sizes based on variations in interference patterns. This paper proposes a hybrid interference-resilient frame fragmentation (Hi-Frag) link-layer scheme for wireless sensor networks. It effectively addresses the challenges associated with dynamic partitioning of blocks while accounting for the observed error patterns. Hi-Frag is the first work to introduce an adaptive frame fragmentation scheme with hybrid block sizing, implemented and evaluated on a real WSN testbed. Hi-Frag shows substantial enhancements over fixed-size partial packet recovery protocols, achieving up to $2.5\times$ improvement in throughput when the channel condition is noisy, while reducing network delays by up to 14% of the observed delay. On average, Hi-Frag shows 35% gain in throughput compared to static fragmentation approaches across all channel conditions used in our experiments. Also, Hi-Frag lowers the energy consumed per useful bit by 66% on average compared to conventional protocols, which increases the energy efficiency.**

Index Terms—**Wireless Sensor Networks, ZigBee, Frame Fragmentation, Interference-Resilient, Energy-Efficiency**

I. Introduction

Frame size has a crucial role in determining the performance of interference-prone wireless networks. However, finding the optimal frame size is a challenging problem. Large frames can achieve good channel bandwidth utilization due to reduction in overheads associated with frame transmission. However, small frames provide better bandwidth utilization in bad channel conditions because of their efficient error recovery. Large frames are used in wired networks for achieving high utilization and throughput because of the low bit-error rates (BER), typically 10^{-15} to 10^{-12}. Wireless networks, on the other hand, have orders of magnitude higher BER, typically 10^{-5} to 10^{-3}. Additionally, the BER in a wireless network can change dramatically from zero to more than 50% over a 1 ms interval [2]. Therefore, there is a need to optimize the trade-off between throughput and error recovery. This is especially critical for resource-constrained wireless sensor networks (WSNs). In practice, TinyOS link-layer has been standardized to 29 bytes, excluding the PHY and MAC layer headers. However, the optimal frame size is dependent on channel quality, which changes with time and environment.

The high BER environment of wireless networks can benefit from packet recovery mechanisms. Frame fragmentation techniques aim to increase the throughput by retransmitting only the corrupted portion of the originally transmitted packet. New data can also be appended to retransmitted portions, thus maximizing channel utilization. Frame fragmentation techniques for wireless communications have focused on the throughput, especially for Wi-Fi and WiMAX technologies. On the other hand, frame fragmentation techniques for WSNs have focused on retransmission reduction as energy efficiency is critical for battery-operated sensor motes.

In this paper, we propose a hybrid interference-resilient frame fragmentation scheme for WSNs called Hi-Frag. The scheme aims to maximize throughput and channel utilization by minimizing retransmissions. Hi-Frag addresses the interference level and pattern in order to achieve better performance. Hi-Frag divides the link-layer frames into blocks of different sizes according to the error patterns. Therefore, the sender can retransmit only the corrupted block instead of the whole frame. We discuss several design choices made to achieve various Hi-Frag goals. We implement Hi-Frag on TinyOS and TelosB motes and compare it with other frame fragmentation protocols under various interference conditions. Our performance analysis focuses on the reduction of retransmission, overhead, and network delay, and on the enhancement of throughput and energy efficiency.

II. Related Work

A large number of partial packet recovery techniques have been proposed in the literature [12], [4], [5], [11]. However, they require hardware modification to the PHY layer which makes them impractical. We limit our discussion to recent frame fragmentation methods that do not need physical layer support since these are the ones similar to our proposed work.

Frame fragmentation techniques could be classified as either static [2], [6] or dynamic [13], [10], [3], based on whether they use fixed or dynamic frame sizes. However, all the available dynamic frame fragmentation techniques need to have an accurate measurements about the link quality such as the channel BER and SNR, which is infeasible for resource-constrained WSNs. Also, they incur high computational and power costs that make them impractical for WSNs.

Seda is one of the early static frame fragmentation techniques [2]. It targets link-layer streaming in WSNs. The protocol divides each frame into equal sized blocks and appends block numbers and CRC fields to each of them.

In case of erroneous transmission, the receiver sends an acknowledgment containing the sequence of the first incorrect block received and a binary block map for the consecutive blocks. Only the corrupted block is retransmitted. One of Seda's limitations is that the loss of acknowledgment forces the sender to resend multiple data frames even if some were received correctly. Another limitation of Seda is the block sizes are fixed which limits the protocol performance (*i.e.*, it cannot adapt to changing channel conditions such as changes in the BER or the interference level). The authors of [6] proposed a similar static fragmentation approach for wireless local networks called fragment-based retransmission (FBR). In FBR, corrupted fragments are given a secondary chance to be transmitted within the same channel access. While the authors mention that either 2 or 4 blocks could be used, they do not describe when to prefer favor one scheme over the other. Moreover, it is not clear how the receiver will know the number of fragments sent by the sender to be able to correctly decode the frame. Finally, the extension of the sender transmission chance could degrade the network fairness. iFrag [8] proposes a dynamic block fragmentation to reduce packet retransmission that uses seda as a baseline. While both protocols use seda as a baseline, there are several key differences between this protocol and iFrag. The design of Hi-Frag allows hybrid block sizes within a frame depending on the perceived error pattern whereas iFrag relies on aggregate error rates to determine the next session uniform block size. Hi-Frag also manages to reduce block overhead by 50% compared to iFrag. The coloring scheme and the tail improve the overall retransmission reduction of Hi-Frag compared to iFrag. More detailed energy consumption analysis is discussed in [1]. In this paper we present the design and implementation of Hi-Frag as well as an overall performance evaluation.

The concept of adaptively changing the size of frame fragments have been proposed in the literature. In [13], an adaptive subpacket scheme that optimizes the block size to maximize throughput is proposed. The size of each block is determined based on the SNR of the channel. Therefore, this technique could not be implemented in WSNs because sensor motes has a limited capability of measuring link quality. Also, the authors never mentioned how to inform the receiver about the newly assigned block size. The author of [10] proposed a segment-based retransmission scheme that use Luby-type erasure code to recover missing symbols. This work is based on the assumption that the transmitter has a precise knowledge of the channel BER and hence will select the segment size accordingly. However, this assumption is not feasible in WSNs. Moreover, the feedback channel is assumed to be error-free and has no delay which is also not realistic.

Noticeably, previous schemes suffer because of three main reasons, namely: high computational cost and low prediction probability, and fixed decisions for all channel conditions. Our scheme, Hi-Frag, proposes a low computation technique that adapts with the channel conditions to reduce packet retransmissions while improving network throughput and delay.

III. Hi-Frag Protocol Design

Hi-Frag is an enhanced partial packet recovery protocol that implements heterogeneous blocks within data frames. The protocol is designed to reduce unnecessary retransmissions and lower the loss rates, leading to higher overall network efficiency. Hi-Frag can better be understood by first looking at the motivations which led to its design and implementation:

1) Dynamic Block Sizes: A block in frame fragmentation technique is a portion of the frame with its own CRC and a block number. This incurs an extra overhead. Dynamic block sizing can reduce this overhead, especially when the channel conditions are good. In addition, a frame can contain varying block sizes instead of homogeneous block sizes provided that both sender and receiver can distinguish the start and end of each block. Hi-Frag supports heterogeneous block sizes in a given frame, gaining the advantage of dynamicity between different frames as well as within a frame.

2) Block Overhead: Block numbers were previously considered necessary to distinguish between out-of-order blocks at the receiver. This overhead can be reduced if the correct sequence can be inferred implicitly. Hi.Frag utilizes a novel technique to eliminate the block number field by including frame number in CRC field calculation before transmission without transmitting it, and implementing a search mechanism on the receiver side to obtain the correct number. This leads to various design modifications. It is necessary to enable the protocol to work properly while maintaining data integrity and reliability provided by 1-byte CRC per block. Frame numbering was used instead of block numbering to reduce the search domain to 4 numbers based on frame order in the current session to preserve the reliability level provided by CRC. A session refers to the frames window a sender transmits before waiting for their ACK to arrive. Therefore, the sender includes the frame number in the calculation of all blocks CRCs within a frame. The receiver decodes the first block's CRC by appending the expected frame sequence numbers in the calculation with data until it finds the corresponding frame. The receiver uses that number with other blocks inside the same frame. In case the the receiver tried all the four numbers without success then the block is assumed to be corrupted, and the protocol searches for the sequence frame number in the next CRC. Because frame number is shared among blocks in a frame, an unexpected error in CRC that leads to incorrect frame number translation is discovered by checking frame number of all correctly received blocks within a frame as well as checking the frame number of previous and next frames. This implementation reliably reduces per-block overhead to 50%.

3) Error Patterns: Observed error patterns due to external interference, especially when caused by other motes implementing the same protocol, may have similarities across successive transmissions. Enabling large data frames containing heterogeneous block sizes is preferred to cope with different error patterns. For example, if the error patterns have high correlation between successive sessions where erroneous bits exist at the beginning of the frame, then it is preferable to send small blocks at the start of the frame and larger blocks near the end of the frame. To enable hybrid block sizes within a frame, both sender and receiver must be able to differentiate between the start and end of each block within a frame, even when some blocks are corrupted. For instance, if the receiver receives a frame with erroneous blocks, the decision to further partition a specific block should be made at both the sender and receiver, while including the possibility of losing the acknowledgment.

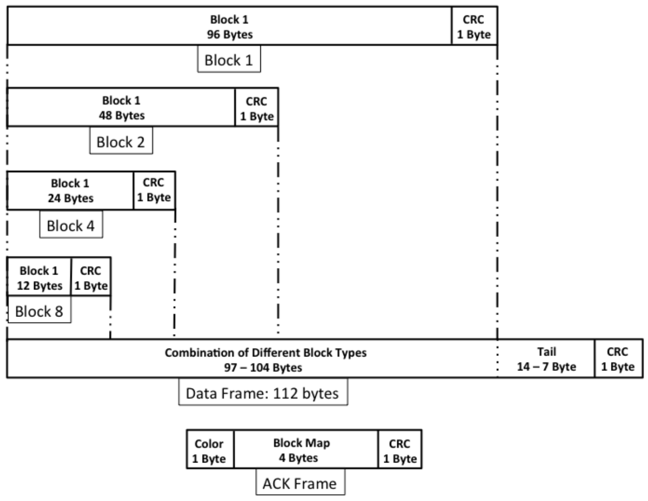

Fig. 1: Hi-Frag frame formats for data and acknowledgment

A. Operation

Hi-Frag defines data frames and acknowledgments (ACKs). The structure of these frames is illustrated in Fig. 1. Hi-Frag frame size is fixed to the largest frame payload supported by the hardware (TelosB), which is 112 bytes. 16 bytes are used by MAC and PHY layers. Each frame contains blocks of different sizes and a tail filling the remaining bytes with data. The size of the tail varies from 7 to 14 bytes, depending on the hybrid blocks within each frame. Hi-Frag uses ACK frames sent after each session to identify the status of blocks. ACK frames consist of a 4-byte block map, one 'Color' byte and a CRC. Both sender and receiver recalculate the frames formats for next session based on the status of the previous session recorded in the block map. Thus, sender and receiver will always have identical formats for the next session.

The operation of Hi-Frag follows the steps shown in Algorithms 1 and 2 and is also highlighted in the flowchart in Fig. 2. At the beginning, both the sender and the receiver agree on the supported block modes during the neighborhood discovery phase. The sender transmits a specific number of consecutive frames each session (4 frames in our implementation), and waits for the ACK from the receiver. At the end of the session, the receiver sends an ACK containing the status of all blocks. Either the block is received correctly, or is corrupted and needs a retransmission. The ACK frame may be sent multiple times until the sender receives it and the next session is started. This design choice was made because losing the small ACK frame

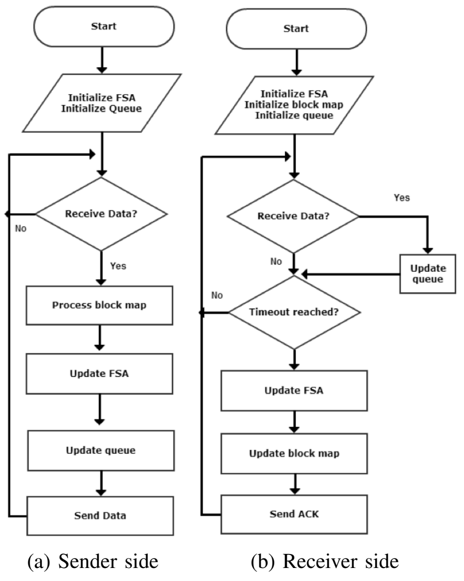

Fig. 2: Hi-Frag operation flow charts

Algorithm 1: HI-FRAG SENDER

```
1  Initiate connection and inform receiver of supported modes
2  Divide network layer packet into blocks
3  Arrange bytes into smallest blocks as specified by Frame Structure Algorithm
   (FSA)
4  Add CRC to blocks and tail, reframe and handoff packet to MAC layer for
   transmission
5  if ACK received then
6      Update FSA:
7      for all blocks check status: do
8          if Block corrupted && size not smallest then
9              Divide into two smaller size
10         else if Block correct && aligned && next block correct then
11             Merge two consecutive blocks into larger block
12         else
13             Use the same block
14     Construct Tail to reach 112 bytes Retransmit missed bytes determined
       from BlockMap and transmit new bytes
15 Continue
```

indicates a noisy channel, and thus it is better to resend a small ACK than retransmitting the large data frame. Also, if the sender retransmits the previous session data, there is a high probability that at least some of the blocks have already been received correctly, and retransmitting them incurs a waste.

One major challenge in the design is to determine each session's frames structure. Blocks can be divided into smaller blocks or aggregated into larger blocks. This introduces additional overhead to determine the suitable structure for the next session. Additionally, both the sender and receiver should agree on the structure to avoid inconsistencies. To achieve these goals, the sender and receiver nodes implement the same Frame Structure Algorithm (FSA) mechanism and share the status of all blocks in each session.

One of the novel techniques introduced in our protocol is the 'Color' field. It serves as an indication that the ACK dedicated for the previous session is being received again, hence there is no need to recalculate hybrid patterns. Duplicate

Algorithm 2: HI-FRAG RECEIVER

```
1  Connection establishment (know sender supported modes)
2  Send ACK including Color and BlockMap to request for frames
3  if data frame received then
4      Check frame number: try from expected until last Update FSA:
5      for all blocks check CRC: do
6          if Block corrupted && size not smallest then
7              Divide next session block into two smaller size
8          else if Block correct && aligned && next block correct then
9              Merge two consecutive blocks into larger block for next session
10         else
11             Use the same block for next session
12     Check tail correctness if all blocks are correctly received then
13         Re-assemble blocks into a network layer packet and handed-off
14     else
15         Buffer correctly received blocks
16 Update BlockMap and Color if session timeout is fired or four frames received
   then
17     Send ACK
18 Stop sending ACK when End Message received or when no new data has been
   received for the period of $timeout_{end}$
```

ACK reception is mainly caused by the receiver assuming that previous ACK was not received, whereas sender did receive it and sent a session which was totally lost. This Color bit is flipped each time the receiver receives data from sender, and is sent in the ACK to the sender. The sender keeps track of the previous color and if there was no change then it assumes all four frames were lost which means that the receiver did not recalculate a new frame structure. In addition to this, 4 bits from color are used as an acknowledgment to the data blocks carried in the tail of each frame. We observe that Hi-Frag ACK frame is smaller than the recent static frame fragmentation approaches, which provides lower expected overhead when the same structure is used and give it higher proportionality to be delivered correctly.

IV. PERFORMANCE EVALUATION

A. Experimentation Setup

Our experimental setup uses TelosB [7] motes in an office environment. The motes use Chipcon-CC2420 radio [9] compatible with IEEE 802.15.4 (ZigBee) standard with data rates up to 250 Kbps. These motes operate in the 2.4 GHz ISM band which interferes with existing WiFi devices. Hi-Frag is implemented in TinyOS 2.1.1.

The experimental setup consists of a sender mote and a receiver mote separated by a distance of $1m$ and powered with USB cables to avoid low battery power. MAC-layer automatic CRC was disabled to allow the reception of partially corrupted packets. Each experiment was repeated under two interference settings. The first set of experiments did not impose any interference. The second set of experiments was done while transferring a large file between two Linux machines equipped with IEEE 802.11g wireless cards within a 15m distance using a transmit power of 15 dBm. We used Wi-Fi as a source of interference because (*i*) it is the main interference source in many WSN deployments such as smart buildings and traffic control applications, and (*ii*) its high power characteristics. We anticipate that our results will hold in environments with lower level of interference.

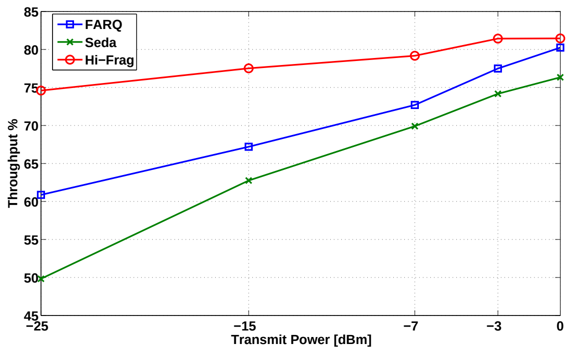

Fig. 3: Throughput results for Hi-Frag, Seda and FARQ with various power level in normal conditions

Term	Meaning
M	The maximum supported number of blocks in a frame
H	Frame header size (bits)
F_L	Frame length (bits)
F_{rec}	Number of received frames
F_{sent}	Number of sent frames
B_L	Block length (bits)
B_H	Block overhead length (bits)
B_{rec}	Number of received blocks
T_{rec}	Received tail length (bits)
ACK_L	Acknowledgment frame length (bits)
ACK_{sent}	Number of sent acknowledgment frames

TABLE I: Terms used in the throughput equations

B. Experimental Results

We evaluate Hi-Frag against Seda and FARQ protocol. Several experiments were carried out with various interference and transmit power setups. In each run, the sender mote transmits 1000 frames to the receiver. Results are then averaged over 5 runs. The average throughput calculation considers all the overhead observed throughout the connection. Equations below show the method used to evaluate throughput of the three schemes (T_{FARQ}, T_{SEDA}, and $T_{Hi-Frag}$). The terms used below are described in Table I.

$$T_{FARQ} = \frac{F_{rec} * (F_L - H)}{(F_{sent} * F_L) + (ACK_{sent} * ACK_L)} \quad (1)$$

$$T_{Seda} = \frac{B_{rec} * (B_L - B_H)}{(F_{sent} * F_L) + (ACK_{sent} * ACK_L)} \quad (2)$$

$$T_{Hi-Frag} = \frac{\sum_{m=1}^{M} (B(m)_{rec} * (B(m)_L - B_H)) + \sum_{i=1}^{F_{rec}} T(i)_{rec}}{(F_{sent} * F_L) + (ACK_{sent} * ACK_L)} \quad (3)$$

In Fig. 3, the throughput of Hi-Frag is compared with Seda and FARQ under normal channel conditions without imposing pathological interference patterns. To make a fair comparison, FARQ implementation is similar to Seda except that it uses a single block per frame. Hi-Frag shows the best throughput performance among Seda and FARQ. Hi-Frag outperforms

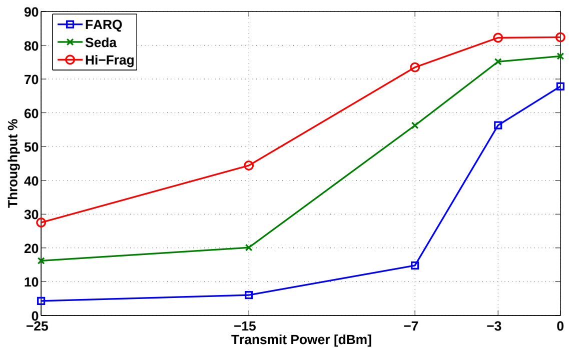

Fig. 4: Throughput results for Hi-Frag, Seda and FARQ with various power level with external interference imposed

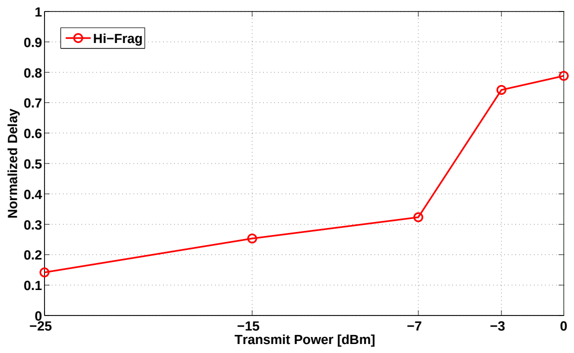

Fig. 5: Average delay of transmitting 1000 frames in Hi-Frag normalized to the delay experienced in Seda

Seda with an average of 20% throughput improvement across all power settings. Also, Hi-Frag outperforms FARQ with an average of 15% because Hi-Frag send slightly more extra bytes as a tail in all channel conditions. However, their performance became very close in perfect channel conditions and sending with highest transmitting power. FARQ results is better than Seda in all transmitting powers due to its lower overhead in good channel conditions.

Fig. 4 shows throughput measurements while imposing strong Wi-Fi interference. Hi-Frag maintains higher throughput compared to Seda and FARQ. Hi-Frag shows up to 2.5× increase in throughput compared to Seda in bad channel conditions. On average, Hi-Frag outperforms Seda by 35%, and FARQ by 150%. This is because Hi-Frag adjusts frame sizes dynamically to minimize the overhead by selecting the appropriate hybrid block sizes in each data frame. As expected, FARQ performs worse than both Hi-Frag and Seda due to retransmissions caused by the interference. One of the major improvements that allow Hi-Frag to outperform Seda and FARQ is the choice of resending small ACK frames instead of sending duplicate data. The interference that caused data frames in the previous session to be corrupted is likely to cause additional frames to be corrupted too. Any additional data frames are overhead, hence degrading network goodput dramatically. ACK frames are about 5× smaller than the average block size, thus have higher probability to be received correctly. If they are not received correctly, then there is a even a lower probability for actual blocks to be received. Additionally, ACK frames in Hi-Frag contain the exact blocks that were not correctly received previously, allowing it to eliminate duplicate block reception entirely. This mechanism also makes Hi-Frag more energy-efficient by reducing the unnecessary retransmissions at the sender.

Fig. 5 compares the overall delay required to send 1000 frames in Hi-Frag versus Seda. The figure shows the normalized network delay under different channel qualities by reducing the mote's transmit power. Hi-Frag outperforms Seda when the interference is increased. The performance peaks at -25 dBm where Hi-Frag achieves a delay of only 14% of that in Seda. It is interesting that Hi-Frag improves delay over Seda even though it was not the main target of the design. This is

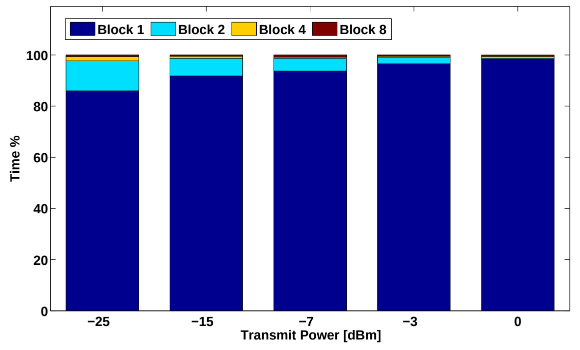

Fig. 6: Percentage of time Hi-Frag spends in each mode without imposed interference

because the receiver in Hi-Frag retransmits the ACK when it lost or corrupted. On the other hand, Seda design make its sender retransmits the entire data frame. Hi-Frag sender waits for a small correct ACK frame to continue transmitting data while a Seda sender waits until a timeout before sending the same data frame that have been sent in previous session if ACK frames are not received correctly.

Fig. 6 shows the percentage of time Hi-Frag spent in sending data with various block types without imposed interference. Hi-Frag's fast transitions allow it to spend less time in undesirable modes at the risk of losing larger data blocks if the interference pattern is independent between consecutive sessions. It is also observed that there is a small number of 8 blocks in all transmit powers. This is because Hi-Frag starts by having a 8-blocks frame structure and quickly recovers after three sessions. Fig. 7 shows the percentage of time Hi-Frag spent in sending data with various block types with interference imposed. Hi-Frag spends significantly longer time in lower block number schemes (larger size) with the existence of interference. However, it is more optimistic as it changes more frequently. It is worth mentioning that Hi-Frag is highly dependent on the interference pattern observed in each run.

We performed more experiments to study the impact of Hi-Frag on power consumption. Our energy calculation method considers only the power consumed by radio transmission and reception, i.e., we ignore the energy consumed by CPU, mem-

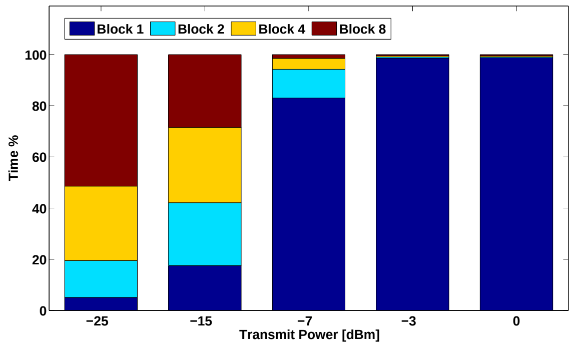

Fig. 7: Percentage of time Hi-Frag spends in each mode with imposed interference

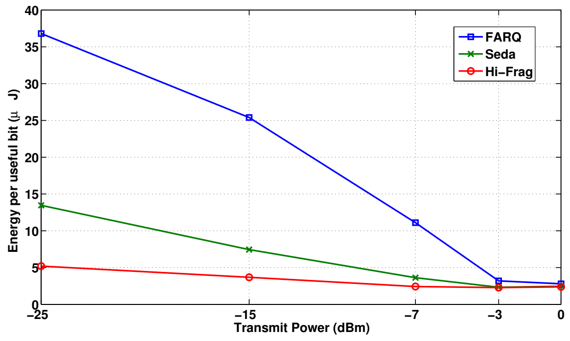

Fig. 8: Energy consumed per useful bit for Hi-Frag, Seda and FARQ with various power level

ory operations, etc, because radio transmission and reception is the dominant source of energy consumption in sensor nodes. Our energy-efficiency comparison considers the amount of energy consumed in delivering a useful bit. We calculate this by dividing the energy consumed in sending and receiving all frames by the number of useful received bits. The useful bits are calculated in a manner similar to that used in our throughput figures. Tx/Rx Energy equals to the transceiver power consumption multiplied by time spent in frames Tx/Rx. We experimentally measure the time spent on sending data and ACK frames over several runs and then calculate the average time to send a single frame for all the compared schemes. Fig. 8 shows the energy per useful bit results in experiments with Wi-Fi interference. Hi-Frag maintains the best energy performance at all transmit powers. Its energy per useful bit results varies between 2.4 and 5.2 μJ. On the other hand, FARQ shows the worst results in these channel conditions. We note that at 0 dBm transmit power, the three schemes yield similar results. This is because at this high transmit power, the number of losses is minimized. However, in environments with less interference, 0 dBm may not always be the optimal transmit power, and considerable energy gains can be obtained by using a reduced transmit power.

V. Conclusions

In this paper we proposed a hybrid interference-resilient frame fragmentation link-layer scheme named Hi-Frag. We presented the various design considerations behind Hi-Frag that improve packet recovery mechanisms in order to reduce packet retransmission and overall overhead. The protocol accounts for highly correlated error patterns in the channel as well as maintaining comparable performance when error patterns are not highly correlated. It also manages to reduce block internal overhead by 50%.

The proposed protocol Hi-Frag implementation and performance was compared and evaluated with Seda and FARQ protocol. Hi-Frag shows up to 2.5× increase in throughput compared to Seda in bad channel conditions. On average, Hi-Frag outperforms Seda by 35% in high interference situations, while maintaining an average of 20% better performance in low interference environments. Hi-Frag maintains the lowest delay in low interference conditions and in highly correlated interference. Moreover, Hi-Frag shows an average of 66% better energy consumption than conventional protocols.

References

[1] A. Daghistani and B. Shihada. Green-frag: Energy-efficient frame fragmentation scheme for wireless sensor networks. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*, pages 458–463, Oct 2013.

[2] R. K. Ganti, P. Jayachandran, H. Luo, and T. F. Abdelzaher. Datalink streaming in wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, SenSys, 2006.

[3] Y. hua Zhu, H. Xu, K. kai Chi, and H. Hu. Accumulating error-free frame blocks to improve throughput for IEEE 802.11-based WLAN. *Journal of Network and Computer Applications*, 35(2), 2012.

[4] K. Jamieson and H. Balakrishnan. PPR: partial packet recovery for wireless networks. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM, pages 409–420, 2007.

[5] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard. Symbol-level network coding for wireless mesh networks. In *Proceedings of the ACM SIGCOMM conference on Data communication*, pages 401–412, 2008.

[6] C.-F. Kuo, H.-W. Tseng, and A.-C. Pang. A fragment-based retransmission scheme with QoS considerations for wireless networks. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing*, IWCMC '07, pages 225–230, 2007.

[7] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Information Processing in Sensor Networks*, 2005.

[8] A. Showail, A. Elrasad, A. Meer, A. Daghistani, K. Jamshaid, and B. Shihada. ifrag: Interference-aware frame fragmentation scheme for wireless sensor networks. *Wirel. Netw.*, 20(7):2019–2036, Oct. 2014.

[9] Texas Instruments. Chipcon CC2420 Datasheet. http://www.ti.com/.

[10] A. Willig. Memory-efficient segment-based packet-combining schemes in face of deadlines. *IEEE Transactions on Industrial Informatics*, 2009.

[11] G. R. Woo, P. Kheradpour, D. Shen, and D. Katabi. Beyond the bits: cooperative packet recovery using physical layer information. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, MobiCom, pages 147–158, 2007.

[12] J. Zhang, H. Shen, K. Tan, R. Chandra, Y. Zhang, and Q. Zhang. Frame retransmissions considered harmful: improving spectrum efficiency using micro-ACKs. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, Mobicom '12, pages 89–100, 2012.

[13] Y. Zhou and J. Wang. Optimum subpacket transmission for hybrid ARQ systems. *IEEE Transactions on Communications*, 2006.