Hands on IRIS: Lessons learned from implementing a cross layer protocol stack for WSNs

Alessandro Camillò
Dept. of Computer Science,
University of Rome, La Sapienza
Rome, Italy.
Email: camillo@di.uniroma1.it

Chiara Petrioli
Dept. of Computer Science,
University of Rome, La Sapienza
Rome, Italy.
Email: petrioli@di.uniroma1.it

***Abstract*—This paper describes the challenges behind the implementation of a full cross-layer protocol stack for wireless sensor networks. Our solution, IRIS, integrates *interest dissemination* and *convergecasting*, adaptive duty cycle and data fusion. The interest dissemination protocol is used for task assignment and to build and maintain the network topology, while convergecasting implements data gathering at the network sink. Convergecasting heavily exploits cross layering in that MAC and routing operations are performed jointly and relay selection is based on flexible cost functions that take into account information from different layers.**

We have implemented and tested IRIS on two different testbeds, in a controlled lab scenario and in a building with Wi-Fi interference. The flexibility introduced by the IRIS cross layer approach results in higher robustness and reliability than that of well-known approaches such as BoX-MAC combined with the CTP protocol. IRIS delivers more than 95% of packets while other approaches have packet delivery ratios as low as 87% and are not as scalable with traffic. IRIS is also able to support timely and robust delivery of data with duty cycles as low as 5%.

I. Introduction

Wireless sensor networks are an emerging technology enabling a host of applications such as environmental monitoring, precision agriculture, surveillance and border control, structural health monitoring, home networking and energy efficient buildings. Recent solutions have clearly shown that cross-layer optimizations can lead to significant improvements in terms of packet delivery ratio, network lifetime and end-to-end latency [1]–[4]. However, such performance improvements have to be proven through extensive experimental campaigns as link instability, impairments associated to signal propagation and hardware limitations affect the reliability and performance of communications in field [5].

This paper describes the challenges behind the implementation of a full cross-layer protocol stack on TMoteSky test-beds. Specifically we have implemented and experimentally evaluated IRIS [2]. IRIS is a cross-layer protocol for WSNs that integrates different desirable features: awake/asleep scheduling, interest dissemination, cross layer convergecasting, data fusion, adaptive duty cycle and estimation of the number of neighbors of each sensor node. In a cross-layer approach each layer of the protocol performs its tasks with the help of other layers. For instance IRIS interest dissemination and convergecasting benefit from information provided by the density estimation module. Relay selection is also performed exploiting a cross layer cost function, favoring aggregation of data and improving overall end-to-end performance.

The authors would like to thank the Wayne State University and University of Padova for letting us use their testbeds. This work has been partially supported by the FP7 ARTEMIS CHIRON project.

To assess IRIS effectiveness in supporting robust and efficient communications in a real setting we have implemented it on TMoteSky nodes and run it on both the Wayne University NetEye testbed and the SIGNET lab testbed. The former is a lab test-bed of up to 130 nodes while the latter is a building level testbed of around 350 nodes. Our experimental validation has the objective to assess the following protocol feature:

- **Reliability**: the protocol stack should deliver a satisfactory percentage of the generated data packets, regardless of the scenario and of instability of the wireless channel.
- **Robustness**: the protocol stack should cope with network topology dynamics, link asymmetry, interferer, etc.
- **Efficiency**: the protocol stack should achieve these two goals with a minimum amount of energy and radio transmissions.

Achieving these goals requires an accurate estimation of the link quality. In the literature stable and agile link quality estimation is usually implemented by means of periodic beaconing. In this paper we demonstrate that it possible to achieve good results, even in highly interfered scenarios, by means of a reactive on demand approach which exploits only the protocol control messages. The resulting protocol stack is shown to be lightweight and robust to interference and varying link quality, being always able to deliver at least 95% of the generated packets with nodes operating with duty cycles as low as 5%.

The paper is organized as follows. In Section II we describe the IRIS protocol stack. Section III provides a detailed description of IRIS implementation. It also identifies the main challenges in developing robust and efficient wireless network protocols. Section IV presents a comprehensive experimental evaluation of IRIS on the two testbeds. Section V discusses related literature highlighting how IRIS improves over previous work. Finally Section VI concludes the paper.

978-1-4673-0921-9/12/$31.00 ©2012 IEEE

II. IRIS: Protocol description

The IRIS stack is composed of several modules each performing its own task. The main modules are: Density Estimator, Duty Cycle management, Interest Dissemination, Convergecasting, Adaptive Duty Cycle and Data Aggregation.

The density estimator module computes the number of active neighbors of a node. This value is subsequently used by the interest dissemination module to determine the number of nodes which need to receive the interest message. The dissemination module provides a network level broadcast service while at the same time allows each node to estimate its hop count distance from the sink. To route the data packets back to the sink the convergecasting module uses the hop count values distributed during the dissemination process. The best next hop relay is selected based on a cost function which combines local parameters such as: hop-count, link quality, queue size, etc. All the packets received are temporary stored inside a FIFO queue and scheduled to be transmitted at the next possible opportunity. The adaptive duty cycle module, interacting with the MAC layer, dynamically adjusts the working duty cycle, temporary switching it to a higher value whenever it detects traffic in the network. Finally the data aggregator module, together with the queue module, allows a node to temporary behave like an aggregator accepting a large number of data packets and merging them together before retransmission. This significantly reduces redundant transmissions, thus network traffic, and improves overall end-to-end performance.

A. Sleeping Modes

For energy saving purposes, every node turns on and off its radio transceiver according to a wake-up schedule that is independent of that of the other nodes (asynchronous duty cycling). IRIS implements two policies of duty cycle: Normal Sleeping Behavior (**NSB**) and Aggressive Sleeping Behavior (**ASB**). More specifically, in NSB time is divided into duty cycle periods of T_n seconds. At the end of each duty cycle period a given node randomly picks a number $t_n \in [0, T_n(1-d_n)]$ where d_n is the normal duty cycle. The node then sleeps for the first t_n seconds (*sleeping period*), after which it wakes up and remains awake for $T_n d_n$ seconds. It then sleeps again until the end of the duty cycle period. The ASB mode happens when a node in NSB does not participate in any channel contention for T_a seconds. In ASB mode a node uses a duty cycle $d_a \ll d_n$. A node in ASB promptly switches back to NSB if it detects any channel activity (*fast wake up*).

B. Interest Dissemination and Density Estimation

In order to perform interest dissemination we use the Fireworks protocol [6]. The procedure is initiated by the sink by transmitting an interest packet to all its neighbors. Whenever a node receives a new interest packet, it rebroadcasts it to all its neighbors with probability p, while with probability $1-p$ it sends it only to c randomly selected neighbors. The main challenge of applying Fireworks within IRIS stems from the fact that a node does not know its neighbors and that, since some neighbors may be asleep, it cannot be sure that the interest packet has reached all its intended destinations. This is why we implemented Fireworks jointly with our online density estimation technique.

The dissemination protocol is executed in rounds, separated enough in time so that the sets of sampled active nodes for each round are statistically independent. In each ith round a node broadcasts a REQ packet, piggybacking in such packet the interest payload and a window size w. Its active neighbors reply with a RES (response) packet in a randomly chosen $[0, w-1]$ slot of the window. The node then collects, and temporary stores, the identifiers of the k_i neighbor nodes which have not yet replied to its REQ. Based on a maximum likelihood estimator, the estimated number of neighbors n_{MLE} of the node based on R samplings is computed as:

$$n_{\mathrm{MLE}} = \left\lfloor \frac{\sum_{i=1}^{R} k_i}{1-(1-d)^R} \right\rfloor \tag{1}$$

where d is the duty cycle.

The new estimate is then used in the next interest dissemination to compute the number of neighbors which should receive the interest according to Fireworks. n_{MLE} is also used to compute w which is set to

$$w = \lceil n_{\mathrm{MLE}} \cdot d \rceil \tag{2}$$

We observed in practice that within a small number of rounds the estimator converges to the real number of neighbor nodes. [1]

C. Convergecasting through cross layer MAC and Routing

Packets are routed towards the sink by exploiting their distance in hops from it. Hop counts are propagated and updated by every node during the interest dissemination phase. Nodes also exploit packet snooping to improve their hop count estimate. Each exchanged packet includes the sender hop count which is then used by the receiving or overhearing node for updating its hop count value. Specifically, each node sets its hop count to the minimum hop count value included in packets it has received plus one.

Assume that a node i has a data packet to send and that the hop count (HC) of this node is n. We define $\mathcal{N}_i(n)$, $\mathcal{N}_i(n-1)$ and $\mathcal{N}_i(n+1)$, $n \in \mathbb{N}^+$, as the sets of neighbors of node i with HC equal to n, $n-1$ and $n+1$, respectively. In IRIS node i elects the next hop relay among the nodes belonging to sets $\mathcal{N}_i(n-1)$ and $\mathcal{N}_i(n)$. Each node calculates a normalized local cost C and uses it for channel contention. Node costs are computed only based on local information, which, for instance, may be related to the state of their battery, the link quality, the queue size, etc. Costs are used during channel contention to locally discriminate among the neighboring nodes that are awake so as to select the most suitable one among them as relay (i.e., the one having the minimum overall cost).

[1] It may happen that an overestimation of the number of neighbor makes a node attempting to transmit interest multiple times without increasing the number of reached nodes. The protocol addresses these situations by means of a timeout

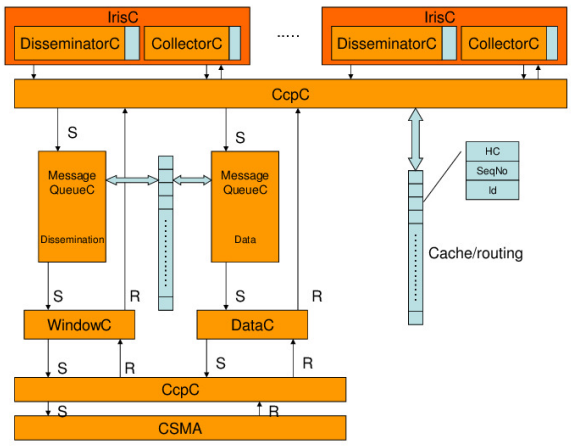

Fig. 1. IRIS architecture

D. Data transmission in bursts

The RTS sent by node i for the selection of a relay, also includes the number b of packets that the sending node is ready to transmit. Upon reception of this RTS, each node j responds (after a time jitter depending on C) with a CTS including the number of packets $b' \leq b$ that this node is willing to receive (this value depends on the available buffer at the receiver). Node i selects the actual number $b'' \leq min\{b, b'\}$ of packets to send back-to-back to the selected relay and unicasts such packets to j which then acknowledges them.

E. Data Aggregation

The deployment area is divided into a number of sectors, each labeled with a given identifier (sector id). At the time they are created, packets are marked with the id of the corresponding sector and nodes can aggregate packets if they have the same sector id. Data aggregation works as follows. Any time a given node receives a packet that can be aggregated with some others in its own buffer (same sector id), it does so creating a single packet where it stores the mean, variance, minimum and maximum of these data. The number of data items aggregated together is also stored in the packet so that it can be aggregated with further data in the future. After that, this node becomes an *aggregator*. Its duty cycle is set to one and a timer, T_{DA}, is started. Until the expiration of this timer, the aggregator remains awake, answering each RTS in the first available slot, so as to promote data aggregation. When the timer expires the node switches back to NSB and inserts a single packet, containing the aggregated data (if any), in its transmission queue.

III. Protocol Details

This section describes the implemented IRIS modules for interest dissemination and convergecasting. It also discusses the problems observed during real life experimentation.

A. Iris Architecture

IRIS has been implemented on TmoteSky wireless sensor nodes running TinyOS 2.x. TmoteSky nodes are equipped with an IEEE 802.15.4 compliant radio transceiver (a ChipCon CC2420 [7] radio chip working in the 2.4GHz frequency band), an MSP430 16 bit micro-controller with 10KB on chip RAM, 48KB FLASH/ROM and 1024KB external FLASH memory. The protocol stack is composed of several modules, as shown in Figure 1, each performing a specific task.

The IRIS data management is based on the concept of *interest*. An interest is a unique identifier, automatically generated by the framework, for a type of data the application is interested in monitoring. This approach provides strong data type checking at compile time (the framework generates prototyped software interfaces) and efficient memory storage allocation.

The convergecasting module is also bound to the unique interest identifiers. In this way it is possible to establish different convergecasting trees for different types of interest. Specifically, it is possible to compute different function costs, depending on the interest payload, providing different QoS levels to information flows.

B. Interest Dissemination and Convergecasting modules

The dissemination service is provided by the WindowC module while convergecasting is provided by the DataC module. All packets which need to be forwarded are stored in a common message queue implementing a FIFO strategy. Packets are extracted from the queue whenever lower layers signal a new transmission can start. The queue size depends on the expected data rate. For our scenarios we set the queue size to 20 packets, in order to accommodate internal generated packets as well packets to forward.

The dissemination service has been implemented as described in II-B and [2] except for the back off management. Instead of being doubled each time, the back-off is randomly selected within a fixed size back-off window. This modification is introduced to reduce dissemination latency. By means of extensive in field tests we realized the main challenge for successful dissemination is temporary unavailability of neighbors due to their low duty cycle and not traffic congestion. The initial value of windos size w is set 2.

The Data module provides a one-hop data communication primitive for the convergecasting protocol.

When a node is involved in a data contention the Ccp module analyzes the hop count of the requesting node, the local node hop count value, the queue size and computes a jitter. Such jitter is then used by the Data module to choose the reply slot of the contention window. Relay selection favors nodes with a smaller hop count. In the current implementation nodes with a smaller hop count randomly choose a slot in the first half of the contention window while nodes at the same hop count of the sender choose from the second half. The sink node has a special value of jitter always set to 0. The processing delay introduced by the software is accounted for tuning the parameters of the contention. Specifically the

slot size has been experimentally set to 10ms considering the latency introduced by the software.

C. Link quality assessment

We have further improved protocol performance by means of strategies aimed at increasing packet reception ratio (PRR i.e. 1 - PER) and topology stability. During the experimental evaluation we, indeed, noticed that some runs experienced a much worse performance, in terms of PRR, than other runs executed few minutes earlier. A deep analysis of the phenomenon has shown that some nodes were, sporadically, able to receive packets sent by nodes at greater distance than the average transmission range. Due to this effect packet snooping, used to improve the hop-count statistics, proved to be more a problem than a benefit: sporadic links were strongly unidirectional (the packet reception ratio of these links was highly asymmetric, e.g. experiencing PRR of 75% in one direction and 0% in the other direction) and the node soon became a dead-end. We modified the hop-count computation removing snooping and accepting hop-count information only upon a complete dissemination handshake based on a two way RTS/CTS schema. This solved the problem of asymmetric links.

Due to some sporadic bidirectional links some nodes sometimes under-estimated their hop-count. This prevented from finding relays during contentions. To avoid this problem we tried to correlate the PRR with RSSI and LQI indicators provided by the radio transceiver. The LQI is computed by the transceiver, for each incoming packet, as an average correlation value, based on the 8 first symbols following the start frame delimiter. The LQI value can be looked at as a measurement of the chip error rate. A LQI value of ≈ 110 indicates a maximum quality frame while a value of ≈ 50 is typical of the lowest detectable quality frames [7]. The correlation between link PRR, obtained running a connectivity test, and LQI gave us an indication that good performing links had an average LQI value ≥ 80. We then first imposed packet discarding in case of communications with LQI < 80. Unfortunately further experimentation has shown that this solution is not robust. Indeed a similar approach had been proposed in the literature [8] motivating our first attempt. However there are still unstable links which experience a LQI value > 80. The reason is that the LQI metric is computed based on received packets so that it does not account for transmissions which result in undetected packets.

To assess the link quality, instead of using expensive beaconing and LQI estimation, we therefore exploited the periodic data transmission contentions as probes for link quality estimation. We modified the Ccp module to downgrade the hop-count estimated value of 1 unit after 25% of failed sending attempts, for a maximum of three decrements. This allowed us to converge to an hop-count reflecting current topology as demonstrated in the experimental results section.

The third modification allowed to improve the convergecasting latencies value of a factor of 10. This was obtained by decreasing the on-time period from a value of 200 ms, used during the simulations, to 20ms. The original value was set to reduce the overhead needed to switch on and off the radio to a small percentage of the on time but leads to a too high toll in terms of latency. A shorter on time also does not affect the probability of finding an active neighbor as the asleep time is correspondingly adjusted to maintain the same duty cycle.

As the last major modification we removed, from the implementation, the carrier sensing phase. According to [2] a node needing to transmit a packet needs to listen for the radio channel at least as long as the window size before accessing the channel. In a dense network this makes the listening phase quite long, resulting in high energy consumption. On the other hand the protocol handshake packets are extremely small, less than 10 bytes, and are then very fast to transmit and unlikely to collide in practice. We have then chosen to trade off the energy waste due to collisions and retransmissions with the energy saved by removing the carrier sensing. This optimization proved also to strengthen the protocol stack against radio interference generated by Wi-Fi.

IV. Experimental Results

We performed our first group of tests on the NetEye testbed hosted by the Wayne State University. The testbed is composed of 130 nodes laid on wooden benches in a large room and deployed in a grid like topology. Each Dell Vostro laptop is attached to 6-12 motes. Each mote is powered from the laptop it is connected to (rather than batteries) and all the laptops are connected to the department Ethernet, which eases gathering of metrics of interest and test-bed reprogrammability. The metrics of interest are stored in a cache and flushed to the central server at regular intervals. In order to make the impact of such operation negligible data are flushed only during the off period of the duty cycle, when the node does not take part to protocol operations.

Because the sensors are deployed 60 centimeters from each other we had to set the transmitting power to the lowest possible value (-25dBm) in order to have a multi-hop topology. Setting the sink node to the node in the upper left corner of the room allowed us to build a topology of up to 60 nodes with route lengths up to 3 hops.

We performed several sets of experiments varying the duty cycle $\in \{5\%, 10\%\}$ and the packet transmission rate.

Traffic was generated according to a Poisson process, by far away nodes in the network. Number of sources varied between 20 and 40. The dissemination process effectiveness was assessed by means of the dissemination coverage ratio, expressed as the ratio between the number of nodes reached by the dissemination and the total number of nodes, and the interest dissemination duration defined as the time needed to reach the last node. The convergecasting process was evaluated based on the packet delivery ratio and the end-to-end latency.

A. Network connectivity

Preliminary experiments were performed to analyze link quality variability over time and external interferers. Connectivity experiments have been performed by transmitting beacon

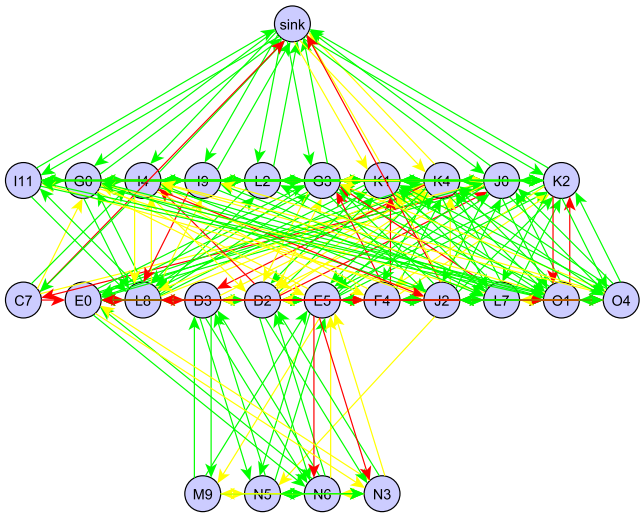

Fig. 2. Typical connectivity graph of the WSN testbed: numbers inside circles indicate node IDs, green lines represent links with packet reception ratios (PRR) greater than 90%, yellow lines represent links with PRR in $[10, 90]\%$ and red lines indicate links with PRR in $[0, 10]\%$. In this graph we show the topology with 20 source nodes

packets over an entire day, every 1 hour at a power of -25dBm. We have computed the packet reception ratio, expressed as the ratio between the number of beacons sent by node A and those received by B, for each pair of network nodes (A,B). Our results indicate that links are subject to substantial variations over time and that asymmetric links often occur. There are indeed nodes located only 60cm apart whose link is highly asymmetric: it has a PRR greater than 90% in one direction and smaller than 10% in the opposite direction.

B. Dissemination

We tested the interest dissemination protocol with $d_n \in \{0.05, 0.1\}$ and d_a = 0.05, with and without the ASB optimization. When active, the ASB module switches the working duty cycle from 0.05 to 0.1 if it hears a packet and maintains the upper value of the duty cycle (i.e., d_n) for a period of 1 second. If channel activity is detected during this period, the timer T_a is reset. Otherwise, after 1s without any detected channel activity, the node switches back to the low duty cycle d_a. Each test we have performed consists of 5 interest disseminations. For each value of the duty cycle we have repeated the experiments several times in order to have results with high confidence level and precision. We have studied the coverage dissemination duration.

Since Fireworks-based interest dissemination is probabilistic, coverage may sometimes not be 100%. We have therefore studied how many interest disseminations are needed to ensure that all nodes receive the interest. By piggybacking the previous interest also in the following interest dissemination we observed that all nodes were always able to correctly receive the interest packet.

As depicted in Figure 3 the dissemination process ends within reasonable time (a few tens of seconds), even at very low duty cycle. When the ASB is active (which means that nodes operate with a very low duty cycle $d_a = 0,05$ until interest disseminaton starts and nodes switch to the normal duty cycle) the interest dissemination duration is comparable to that experienced when operating in normal mode with $d_n = 0,1$. This is by no means a trivial achievement. Indeed, when operating in normal mode with $d_n = 0,05$ the interest dissemination duration doubles. This result shows that the ASB optimization is a key tool to reduce energy consumption while maintaining good latency performance. Gain in energy consumption with respect to NSB depends on the frequency of interest dissemination: the lower the frequency, the closer the energy consumption of ASB is to that of NSB operating at very low duty cycle ($d_n = d_a$). A decrease in energy consumption of 50% with respect to NSB was obtained in our experiments, which lasted 30 minutes each.

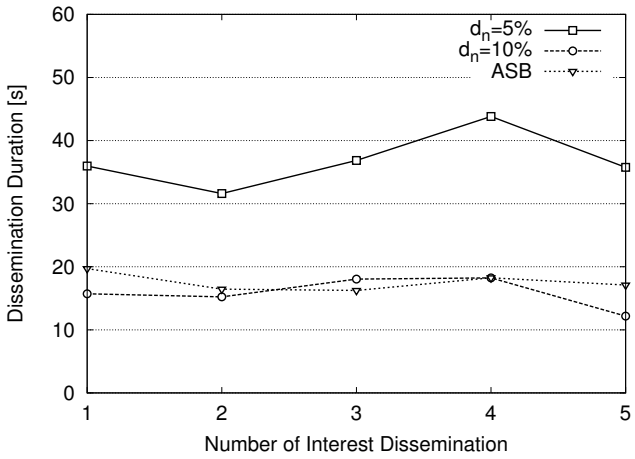

Fig. 3. Interest dissemination duration for $d_n \in \{0.05, 0.1\}$ and ASB with $d_n = 0, 1$, $d_a = 0, 05$

C. Converge casting

The convergecasting protocol has been tested by varying $d_n \in \{0.05, 0.1\}$ in combination with the three optimizations: adaptive duty cycle, burst transmission and data aggregation. We tested the IRIS performances using traffic rates $\lambda \in \{0.25, 0.125, 0.062\}$ $pkts/sec$ and varying the number of data source nodes $\in \{20, 25, 30, 35, 40\}$ [2] each transmitting 100 packets within a time interval of 10 minutes.

Figure 4 shows the average end-to-end latency experienced by selected sensor nodes for different duty cycles, in the WSN topology displayed in Figure 2. All nodes are able to deliver data to the sink within 8s when $d_n = 0, 1$ and within 27s when $d_n = 0, 05$.

Nodes $C7$, $M9$, $N5$ and $N6$ show the worst performance as these nodes are placed farthest away from the sink in the considered WSN topology. Packet delivery ratios, for not congested scenarios, are always greater than 97%. With the

[2]This was achieved adding to the topology in Figure 2 extra source nodes geographically located in the part of the deployment area far away from the sink

Source nodes	Dropped pkt	Cong. nodes
20	0%	0%
25	1.15%	13.5%
30	5.11%	35.16%
35	9.82%	47.72%
40	21.98%	70%

TABLE I
CONGESTION VARYING THE NUMBER OF SOURCE NODE. $\lambda = 0.25$.

highest traffic rate (i.e 40 source nodes and $\lambda = 0.25$) the network is congested and the packet delivery ratio drops down to 70%. The number of packets dropped because of congested nodes is about 22% of the total, whereas 70% of the nodes experience congestion (internal queue full). Table I shows detailed results for the different traffic rates.

IRIS variant without back-to-back (burst) data transmission has much worse performance. For instance the packet delivery ratio is about 31% at the highest rate, because of the congestion: 80% of the nodes are congested with a dropped packet percentage of 51%.

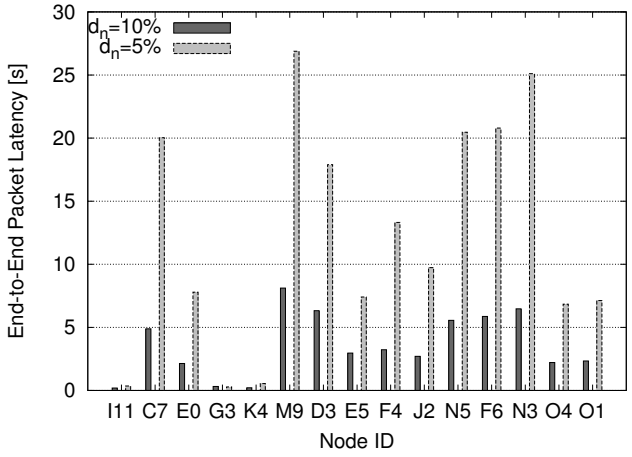

Fig. 4. End-to-end data latency from selected sensor nodes for different d_n.

D. Comparison against existing implementations.

We compared IRIS convergecasting performance with that of TinyOS MAC and routing modules. More precisely, we used BoX-MAC [9] and CTP [10] to route data towards the sink. Comparisons have been performed for $d_n = 0.1$ in the Wayne Un. testbed, in scenarios similar to the one discussed above.

CTP does not support burst transmission. The application needs to wait for a packet to be sent before sending the next packet. This does not allow CTP to sustain a high data rate and its performance drops more sharply than IRIS's as the load increases.

IRIS is also much more robust to interference. To show this we performed experiments in a second testbed hosted by the SIGNET lab at the University of Padova. This testbed comprises 350 wireless sensor nodes deployed across different buildings, transmitting at maximum transmission power.

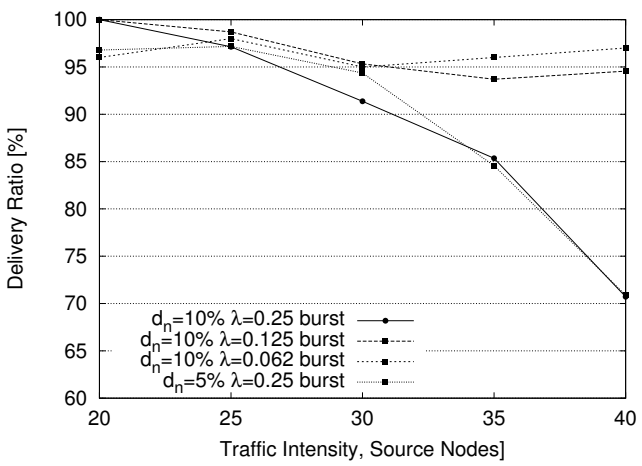

Fig. 5. Delivery ratio when varying the traffic rate $\in \{0.25, 0.125, 0.062\}$ and the duty cycle.

	TinyOS		IRIS	
Source nodes	avg	stdev	avg	stdev
10	93.75	6.95	95.0	9.21
22	90.0	10.0	96.0	4.89
27	90.0	12.24	95.0	5.0
32	93.75	6.95	98.0	6.0
36	97.5	4.33	98.0	4.0
67	87.5	9.68	100.0	0
69	92.5	6.61	100.0	0
252	95.0	7.07	100.0	0
253	92.5	9.68	97.0	4.58

TABLE II
IRIS VS. BOX-MAC AND CTP: PACKET DELIVERY RATIO [%].

Nodes are deployed inside normal office rooms. The department is equipped with WLAN Internet access. The Wi-Fi network used during day-time produces interference which affects the mote network, making the sensors topology quite unstable.

The testbed management system allows to use different sections of the testbed for different experiments running in parallel. Experiments have been performed in a configuration similar to that of the Wayne un. scenario, using 27 sensor nodes scattered across three floors of the DEI/A building. The network topology diameter is 4 hops. All nodes but the sink neighbors are sources, and generate packets at a rate of one packet every three minutes.

Table II shows the packet delivery ratio (PDR) in terms of average (avg) and standard deviation (stdev) performance for selected nodes, comparing IRIS and the solution combining Box-MAC and CTP.

IRIS cross layer approach provides a higher reliability in terms of average PDR and also makes the delivery performance more stable, through a reduction of the PDR standard deviation. Note that while IRIS packet delivery ratio is greater than or equal to 95% in all cases, it can be as low as 87% for BoX-MAC and CTP.

V. Related Work

In multi-hop wireless sensor network scenarios routing plays a key role for designing robust, reliable, efficient and energy efficient systems. Different routing protocols [11] have been proposed in the literature to favour load balancing of traffic among nodes, to improve energy-efficiency, robustness in presence of interference or link dynamics. The majority of such solutions do not make any assumption about the underlying MAC protocol. Integration of off the shelf routing and MAC layer protocols for WSNs is however less trivial than expected, as shown by [12]. In this paper authors wired the implementation of MintRoute [13], a convergecasting protocol, with T-MAC [14] and tested the resulting solution in a large scale deployment. They observed that, despite the implementation was performing satisfactorly well in a controlled lab environment, it suffered from poor performance in field, where many unforeseen events occurred. This paper clearly demonstrates the need to design complete, robust solutions which can adapt to a challenging environment, and the importance of extensive experimentation. Based on in field experience communication stacks should be developed which explicitly account for the most commonly experienced problems, improving WSN robustness and reliability. Among the solutions proposed at the network layer, CTP [10] is the default standard used as benchmark for real life deployments. It uses periodic beacon messages for building and maintaining a routing tree, and data messages to report application data to the sink. To create a stable routing table the collection protocol exploits link quality estimation based on the expected transmission value (ETX). The ETX value of a given link (x, y) is determined by the expected number of transmissions needed for x to unicast a packet to y which is correctly received and successfully acked. Typically, CTP is combined with X-MAC or its variants in order to provide a complete solution for WSN operation. DISSense [15] improves over this approach, with the objective to support adaptive ultra-low duty cycle operation in applications where packet generation is strictly periodic.

A completely different approach to wireless sensor network design has been recently proposed by cross layering solutions [1], [3]. In these approaches relay selection is opportunistically performed, based on a function which expresses the suitability of an active neighor to serve as relay. Routes dynamically change over time depending on the varying link quality, node load, node residual energy. This approach therefore naturally provides the flexible communication service which is needed to handle unexpected events which may compromise the correct network operations or degrade its performance.

In [2] we have shown that IRIS not only provides a reacher communication service than previous cross layer solutions, but it also outperforms, based on simulations, the best performing solutions available in the literature. What is lacking is therefore an extensive experimental study of cross layering approaches and IRIS, to show which are the challenges which could prevent from reproducing in field the extremely good simulation performance of IRIS and cross layering solutions. Filling this gap has been the driving objective of this paper.

VI. Conclusions

This paper describes IRIS protocol implementation and experimental validation. IRIS is shown to be able to provide a timely and reliable communications service for wireless sensor networks, at the same time allowing nodes to operate at a very low duty cycle.

References

[1] P. Casari, M. Nati, C. Petrioli, and M. Zorzi, "Alba: an adaptive load-balanced algorithm for geographic forwarding in wireless sensor networks," in *Proceedings of IEEE Military Communications Conference*. (MILCOM 2006), pp. 1–9.

[2] A. Camillò, M. Nati, C. Petrioli, M. Rossi, and M. Zorzi, "Iris: Integrated data gathering and interest dissemination system for wireless sensor networks," *Ad Hoc Networks*, 2011. Available: http://www.sciencedirect.com/science/article/pii/S1570870511002009

[3] M. Vuran and I. Akyildiz, "Xlp: A cross-layer protocol for efficient communication in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 11, pp. 1578–1591, 2010.

[4] D. Ferrara, L. Galluccio, A. Leonardi, G. Morabito, and S. Palazzo, "Macro: an integrated mac/routing protocol for geographic forwarding in wireless sensor networks," in *Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3. (INFOCOM 2005), pp. 1770–1781.

[5] D. Kotz, C. Newport, and C. Elliott, "The mistaken axioms of wireless-network research," Dept. of Computer Science, Dartmouth College, TR2003-467, July 2003. Available: http://www.cs.dartmouth.edu/ dfk/papers/kotz-axioms-tr.pdf

[6] D. Dubhashi, O. Häggström, L. Orecchia, A. Panconesi, C. Petrioli, and A. Vitaletti, "Localized Techniques for Broadcasting in Wireless Sensor Networks," *Algorithmica*, vol. 49, no. 4, pp. 412–446, Dec. 2007.

[7] C. Datasheet, "2.4 ghz ieee 802.15. 4 zigbee-ready rf transceiver," 2006. Available: http://www.stanford.edu/class/cs244e/papers/cc2420.pdf

[8] "Tinyos multihoplqi routing algorithm." Available: www.tinyos.net/tinyos-1.x/tos/lib/MultiHopLQI/

[9] D. Moss and P. Levis, "Box-macs: Exploiting physical and link layer boundaries in low-power networking," Technical Report SING-08-00, Stanford University, , 2008.

[10] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. (ACM SENSYS 2009), pp. 1–14.

[11] V. Gungor, M. Vuran, and O. Akan, "On the cross-layer interactions between congestion and contention in wireless sensor and actor networks," *Ad Hoc Networks*, vol. 5, no. 6, pp. 897–909, 2007.

[12] K. Langendoen, A. Baggio, and O. Visser, "Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture," in *Proceedings of the IEEE 20th International Parallel and Distributed Processing Symposium*. (IPDPS 2006), pp. 8–pp.

[13] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems.* (ACM SENSYS 2003), pp. 14–27.

[14] T. Van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*. (ACM SENSYS 2003), pp. 171–180.

[15] U. Colesanti, S. Santini, and A. Vitaletti, "Dissense: An adaptive ultralow-power communication protocol for wireless sensor networks," in *Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems*. (IEEE DCOSS 2011), Barcelona, Spain, June 27-29th, 2011, pp. 1–10.