Integrated Data Delivery and Interest Dissemination Techniques for Wireless Sensor Networks

Michele Mastrogiovanni*, Chiara Petrioli*, Michele Rossi†, Andrea Vitaletti* and Michele Zorzi†
*Computer Science Department, University of Rome La Sapienza, via Salaria 113, 00198 Roma, Italy.
†Department of Information Engineering, University of Padova, via Gradenigo 6/B, 35131 Padova, Italy.

***Abstract*— The paper presents IRIS, an Integrated Routing and Interest dissemination System for wireless sensor networks. The proposed protocols are designed to work under very low duty cycle operations and are jointly optimized for improved efficiency. Routing towards the sink is achieved by exploiting hop count information which is proactively distributed during the interest dissemination phase. Node densities are locally and dynamically estimated at each node and exploited at the MAC layer by means of a cost based probabilistic scheme. A cross-layer routing/MAC scheme is defined where relays to the sink are selected based on nodes' resources (including energy and queue occupancy). The proposed solution is a step towards the definition of complete, self-adapting and autonomous sensor network systems.**

I. Introduction

In the last few years, wireless sensor networks have enjoyed an extremely high popularity in the research community [1] [2] [3] [4] [5] [6]. These works range from data dissemination algorithms [3] [4] to channel access techniques [2] [5] [6] [7] as well as interest dissemination protocols [1] [8] [9] and neighbor estimation algorithms [7]. In [1] the authors propose directed diffusion, where interests are first disseminated by flooding the network, and subsequently used to build routes towards the data gathering node (sink). Further work on interest dissemination protocols can be found in [8] and [9], where more refined techniques are proposed to lower the energy consumption and the overhead imposed by flooding without affecting its reliability (successful delivery of the interest packet to the sensor nodes). As an additional tool to increase energy efficiency, in [2] the authors concentrate on the study of a channel access scheme which aggressively exploits the node sleeping behavior. Data delivery protocols (network nodes $\rightarrow$ sink) can be found in [3] and [4], where the authors exploit geographical coordinates by devising integrated MAC and routing protocols for wireless sensor networks with awake/asleep sleeping cycles. Finally, recent papers [5] [6] [7] focus on refined MAC procedures and on the estimation of the number of neighbors (local density) at each sensor node. We note that previous work mainly focuses only on some aspects of the whole system, by either addressing the forward (interest dissemination) or backward (data delivery) communication phases, without considering them together. Integrating these two phases poses some challenges. For example, solutions for interest dissemination such as [9] require that each node knows its neighbors and their awake/asleep schedule in order to work, whereas solutions such as [3] do not require such knowledge.

In this paper we present IRIS, an Integrated Routing and Interest dissemination System for wireless sensor networks where data gathering and interest dissemination operate concurrently and are assisted by neighbor estimation and awake/asleep algorithms for improved efficiency. Towards this end, we consider most of the above issues and related schemes and we integrate them in a coordinated manner. Sleeping cycles are accounted for to prolong the network lifetime, whereas density estimation is carried out to assist the operations performed by both MAC/routing and interest dissemination schemes. The result of our work is a set of on-line, self-starting and adaptable algorithms for interest dissemination and information retrieval in wireless sensor networks.

The paper is organized as follows. We start describing the IRIS framework in Section II. The system is composed of several cooperating schemes, namely, sleeping behavior control (Section II-A), density estimation algorithms (Section II-B), integrated MAC and routing (Section II-C) and an interest dissemination protocol (Section II-D). In Section II-E we outline how these mechanisms inter-work. Section III illustrates the performance evaluation of our approach. We draw our conclusions in Section IV.

II. Data Dissemination Framework

The techniques we propose for wireless sensor networking span from awake/asleep scheduling protocols to interest dissemination methods, MAC/routing schemes and estimation of the number of neighbors of each sensor node. We stress that these techniques operate in an environment characterized by awake/asleep cycles (see Section II-A) and are activated when actually needed. Each sensor communicates with its neighboring nodes in three cases: 1) to send a data packet towards the sink node, 2) to propagate an interest (broadcast communication) and 3) to estimate the number of neighbors (local density). These three tasks are interleaved during the node lifetime and their settings may be dynamically changed according to the node requirements. For instance, we might dynamically decide the length of the interest and density estimation procedures according to the percentage of nodes that we intend to reach and the desired estimation accuracy, respectively. We note that density estimation is needed by all the protocols that we integrate in our framework. Estimation procedures consume resources in terms of time and energy. However, as we demonstrate in Section III, our schemes are robust to estimation errors, which means that estimating the number of neighbors with an error of approximately 50% in many cases suffices to achieve very good performance.

This material is based upon work partially supported by the MIUR International FIRB RBIN047MH9, the EU Integrated Project e-SENSE (IST-4-027227-IP) and the EU Integrated Project AEOLUS (FET-15964). We would like to thank Prof. Stefano Basagni for the insightful discussions during the manuscript preparation.

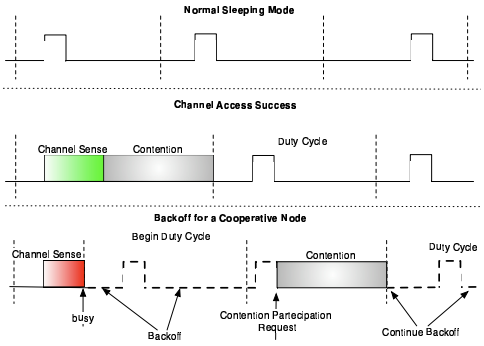

Fig. 1. Nodes sleeping behavior

A. Sleeping behavior

Sleeping modes are implemented to reduce energy consumption and prolong network lifetime. When a node does not have data traffic to send, it follows the so called basic sleeping behavior. According to the basic algorithm, a given node divides time into periods of T seconds (*sleeping cycle* periods). At the end of every sleeping cycle it randomly picks a real number $t_a \in [0, T(1-d)]$, where $d > 0$ is the duty cycle (Fig. 1).

During the subsequent sleeping cycle, the node will sleep for the first t_a seconds, after which it will wake up and remain in the active state (listening to the wireless medium) for Td seconds, and then it will go back to sleep up to the end of the sleeping cycle. Note that sleeping cycles at different nodes are not synchronized. The sleeping mode dynamics are slightly different when a node has data to send. In our work, we adopt a CSMA-based MAC. Hence, before sending its data a node first senses the channel to detect ongoing transmissions. If the channel is sensed idle, then the node starts the channel contention with its active neighbors in order to elect a relay node. The node remains active during the whole channel contention until a relay node is finally elected and the packet forwarded. The contention follows the procedure described in Section II-C. After the packet transmission, the node resumes the basic sleeping procedure. If instead the channel is sensed busy, the node initializes a back-off timer and follows the basic sleeping procedure until this timer expires. At the end of the back-off period, the node performs a new channel sense by repeating the procedure described above. If the channel is still busy, the node enters a new back-off period, where the back-off duration is doubled with respect to the previous attempt. We finally describe the sleeping behavior for a cooperative node. We define as cooperative a node that has data to send, but may accept to relay traffic for a neighbor when it is asked to do so during a back-off period, i.e., after having unsuccessfully tried to access the channel to forward its own data. In this case, the node starts sensing the channel, which is found busy. As above, it starts a back-off timer and resumes the basic sleeping algorithm. If, however, the cooperative node is contacted by a neighbor before the expiration of this timer, it accepts to help the inquiring node and participates in the contention to forward its traffic.

B. Node density estimation

As clarified in the next subsections, for proper operation, many of the techniques considered in our approach require local node density estimates. Algorithms to estimate the number of neighbors can be activated on demand, periodically, or when certain events occur.

In this section, we present an estimation procedure for the total number of nodes within coverage of a given target node. Each node is allowed to turn on and off its radio according to a duty cycle d, which is assumed to be common to all nodes in the network. The problem to be solved is to precisely estimate the total number of nodes within coverage, including *both awake and sleeping* devices. To this end, we implemented an iterative estimation procedure as follows. The estimation algorithm is executed in rounds; for analytical purposes, we assume that rounds are sufficiently separated such that the sets of active nodes which are sampled at each estimation round can be considered to be statistically independent. Note that, due to such an assumption, each node is found to be in the active state in each round with probability d, independently of the past sampling history. At each estimation round the target node counts the number of active nodes within coverage. This can be achieved using known multiplicity estimation algorithms. To this end, we consider here two alternative approaches. The first approach, exploits the Binary Tree Estimation (EBT) scheme proposed in [7]. This scheme uses a binary tree search and allows for both a complete counting of the in-range devices as well as a partial counting. In the partial counting case, the algorithm provides estimates for the total number of in-range neighbors and indications of the estimation errors. Hence, one might use the algorithm to either count *all* active in-range devices or continue until the desired estimation accuracy is reached. In the present work, we consider the first option (complete counting), and leave the investigation of partial counting for future study. The second approach, that we call WINDOW, uses a simple protocol based on a contention window as follows. The inquirer (target node) starts the counting procedure sending a REQ message, which is followed by a window of W time slots. On receiving the REQ, each node randomly picks a slot in $1, 2, \ldots, W$ and replies with a short packet (whose transmission time fits into the slot duration) including its own identifier (id). The interrogator collects the number of successfully transmitted packets in the window and memorizes the ids of the related nodes. For each subsequent estimation round, the window size is taken as twice the current estimate of the number of active neighbors. Note that EBT is more accurate than the window based approach described above as contentions are distributed along binary trees which eventually consider each active neighbor within range. In the window based approach, instead, collisions may always occur even if we increase the window size. Note also that a maximum window size has to be set for practical reasons. However, as shown later, the WINDOW approach for moderate node densities is usually faster while still providing very good results.

We consider that a single estimation procedure lasts a given number of rounds r. At every round $i = 1, 2, \ldots, r$, we count the number of *active nodes* within coverage $k_1, k_2, \ldots, k_r$. We begin the estimation procedure at round 1. At the generic estimation round $i \geq 1$, the target node counts all active nodes that, however, have not been yet counted in rounds $1, 2, \ldots, i-1$ (previous rounds). This is implemented as follows. Each node must reveal its presence to the target (inquiring) node in the first round $i \geq 1$ in which the node is in the *active state* and then must stay silent for all future requests belonging to the

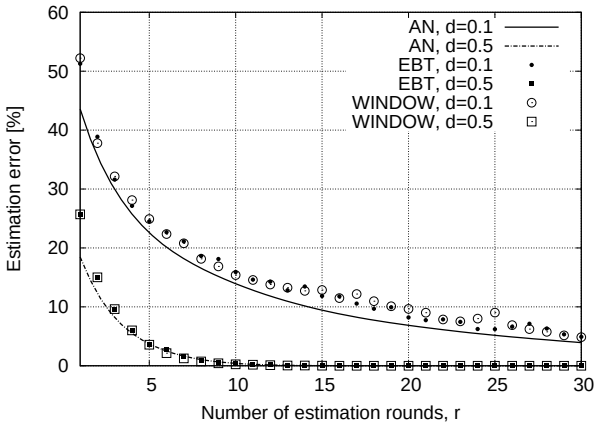

Fig. 2. Comparison of estimation procedures

same estimation procedure (rounds $i+1, \cdots, r$). Hence, at every round $i = 1, 2, \ldots, r$, the target node detects a number of active nodes $k_1, k_2, \ldots, k_r$. Note that k_is do not contain repetitions. That is, k_i will contain all nodes that were active in the i-th round but that were not active in all previous $i-1$ rounds. After r estimation rounds, we can use this information to construct a likelihood function as follows

$$L(n, k_1, \ldots, k_r, d) = \prod_{i=1}^{r} \binom{n - S(i)}{k_i} d^{k_i} (1-d)^{n-S(i)-k_i} \quad (1)$$

where n is the total number of nodes within coverage that we need to estimate and

$$S(i) = \begin{cases} 0 & i = 1 \\ \sum_{j=1}^{i-1} k_j & i > 1 \end{cases} \quad (2)$$

The maximum likelihood estimate for n, which we call $\tilde{n}$, is finally found as $\tilde{n} = \arg\max_n L(n, k_1, k_2, \ldots, k_r, d)$.

As an example, in Fig. 2 we report the average estimation error by considering the above analytical approach (AN) and ns2 [10] simulation of the EBT and WINDOW counting procedures. The results are plotted for two typical values of the duty cycle $d = 0.1$ and $d = 0.5$ and confirm the validity of the estimator and the goodness of both counting methods. Further results are given in Section III.

C. MAC and routing algorithm

For the routing, we implemented SARA (Statistically Assisted Routing Algorithm), proposed in [11]. Packets are routed towards the sink node by exploiting hop count (HC) topologies.[1]. Hop counts are propagated and updated by every node during the interest dissemination phase according to a procedure similar to the one in [9], where HC packets are disseminated according to back-off intervals as in [12] (to minimize the collision probability and most importantly to reduce the number of hop count estimates which need to be transmitted by each node). As in [11], routing is modeled as a sequential decision process, where at every decision stage a node has to take a specific action, i.e., to select the best relay node for the current transmission.

[1]Hop counts are defined as the minimum number of transmissions to reach the sink from a given node when all nodes are awake.

Assume that the currently occupied node is node i, that its hop count is $\text{HC}(i) = n$ and that the forwarding process is at stage $t \in \mathbb{N}$, where time evolves one unit every decision stage (i.e., every forwarding action). 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. The problem to be solved is to decide which is the best relay among the nodes in sets $\mathcal{N}_i(n)$ and $\mathcal{N}_i(n-1)$. Nodes in set $\mathcal{N}_i(n+1)$ are not considered as they very unlikely lead to satisfactory solutions [11]. In addition, at the current node i, we associate a (normalized) cost $c_j \in [0, 1]$ to each link (i, j), $j \in \mathcal{N}_i(n-1) \cup \mathcal{N}_i(n)$. These costs may be related to queue lengths (congestion), node residual energies, link states in terms of success probability, etc. The specific cost function that we consider in our implementation is discussed in Section III. We refer to $j_{n-1}^t \in \mathcal{N}_i(n-1)$, $j_n^t \in \mathcal{N}_i(n)$ and to c_{n-1}^t, c_n^t as the minimum cost nodes in sets $\mathcal{N}_i(n)$ and $\mathcal{N}_i(n-1)$ and their associated costs, respectively. We further define *forwarding cycle* as the sequence of steps between the forwarding stage where a node with hop count n is reached for the first time and the stage where a neighbor with hop count $n-1$ is eventually selected as relay. That is, a cycle is the number of stages it takes the packet to advance one hop towards the sink. We note that, in order to minimize the delay, the optimal choice would be to always forward the packet towards node j_{n-1}^t. However, when the cost of link (i, j_{n-1}^t) is high, it might make sense to route the node towards node j_n^t, with the hope that this node has a more convenient neighbor with a HC equal to $n-1$. In this way, we actually postpone the hop count advancement ($n \rightarrow n-1$) to the next forwarding step. In mathematical terms, the forwarding option $n \rightarrow n-1$ is preferred when $c_{n-1}^t \leq c_n^t + \mathcal{E}$, where $\mathcal{E}$ is the expected minimum cost among nodes with HC $n-1$ at the next stage $t+1$. In the following, we refine this concept by presenting the online optimal routing policy in our settings. See [11] for further details and a formal proof of its optimality. At every stage $t \geq 0$, a decision has to be made on whether the packet has to be forwarded to node j_{n-1}^t or node j_n^t. The cost accumulated (assuming additive costs) from the beginning of the current forwarding cycle[2] $C_{tot}(t)$ is defined as $C_{tot}(t) = C_{par}(t) + c_{n-1}^t$, where $C_{par}(t)$ is defined as

$$C_{par}(t) = \begin{cases} 0 & t < 1 \\ \sum_{k=0}^{t-1} c_n^k & t \geq 1 \end{cases} \quad (3)$$

The minimum cost of all paths to a node with hop count $n-1$ encountered by the packet from step 0 to step t (the current step) is evaluated as follows

$$C_{tot}^{min}(t) = \min_{0 \leq k \leq t} \left\{ C_{tot}(k) \right\} \quad (4)$$

It can be proven [11] that the online optimal routing policy obeys the following stopping set

$$\mathcal{B}_2 = \left\{ X_t : C_{tot}^{min}(t) - C_{par}(t+1) \leq \mathcal{E} \right\} \quad (5)$$

That is, at time t the packet is routed towards node j_{n-1}^t if the inequality in set $\mathcal{B}_2$ is verified [11]. Next, we summarize

[2] We assume that the current cycle started at time 0.

an integrated MAC/routing scheme, proposed in [13], which exploits the previous routing rule

1) Let the forwarding process currently be at node i, having $\mathrm{HC}(i) = n$. Also, assume that $C_{tot}^{min}(t-1)$ and $C_{par}(t)$ have been computed by this node (note: these values can be forwarded along with the packet).
2) Node i begins the channel contention by transmitting a $\mathrm{REQ}(n-1, \rho = 0, N, \mathcal{T})$ to trigger a reply from all awake nodes in set $\mathcal{N}_i(n-1)$. N and ρ are the estimated number of awake nodes and the estimated cost correlation (initially set to 0) for the nodes in this set, respectively. The REQ contains a tabu list $\mathcal{T}$ carrying the identifiers of the last *tabulen* visited nodes. This list is used to avoid ping-ponging between nodes at the same HC distance n. In addition, the REQ also includes the information needed for the computation of the link cost $c(i,j)$ at the receiving node $j \in \mathcal{N}_i(n-1)$.
3) Every awake node $j \in \mathcal{N}_i(n-1)$ calculates a probability $P_a = P_a(c(i,j), \rho, N)$, where $c(i,j)$ is the cost of link (i,j), and ρ and N are the estimates included in the REQ. P_a is computed by means of the channel access functions in [13]. All nodes $j \in \mathcal{N}_i(n-1)$ reply to the REQ with probability P_a. When $j \in \mathcal{T}$, P_a is set to zero. Node identifier and cost are included in every reply (REP).
4) The following three events can occur: *collision*: multiple nodes in $\mathcal{N}_i(n-1)$ reply to the REQ and no correct reply is detected by node i; *silence*: no nodes reply; *success*: a single node, say node j_{n-1}^*, replies to the REQ. If either *collision* or *silence* occurs, the node i sets $\rho \leftarrow \min(1, \rho + \Delta\rho)$, transmits a REQ including the new ρ, and the channel contention is continued from step 2 above. In case of a successful event, node j_{n-1}^* wins the contention.
5) Node i begins a further channel contention by sending a $\mathrm{REQ}(n, \rho = 0, N, \mathcal{T})$ addressing all awake nodes in $\mathcal{N}_i(n)$. This contention follows the procedure described in steps 2–4 with the only difference that nodes now include the quantity $\mathcal{E}$ in their REPs. We refer to the winner of this contention as j_n^*.
6) $C_{tot}^{min}(t)$ is obtained as $C_{tot}^{min}(t) \leftarrow \min\{C_{tot}^{min}(t-1), C_{par}(t) + c(i, j_{n-1}^*)\}$. The relay node is finally selected by using Eq. (5): node j_{n-1}^* is picked if $(C_{tot}^{min}(t) - C_{par}(t) - c(i, j_n^*)) \leq \mathcal{E}$, otherwise node j_n^* is selected.

The rationale is to shape P_a according to the node costs in order to promote low cost paths. That is, a node participates in the channel contention to be the relay according to its own cost, thereby promoting low-cost paths already in the MAC access phase. The correlation estimate ρ is also accounted for, as the optimal channel access behavior depends on the degree of similarity among node costs [13]. The scheme presented above has been implemented to forward data packets towards the sink, whereas the interest dissemination phase employs a simple CSMA MAC scheme.

D. Interest dissemination

Algorithms for interest dissemination are a fundamental part of the overall network system. In fact, they are responsible for communicating control messages to all nodes such as the type of data to be sent to the sink and the QoS requirements to be met, i.e., how this data should be propagated. This operation usually involves one-to-all communication which is initiated and governed by the sink. However, we observe that broadcasting data in sensor networks may be expensive and, at the same time, challenging. In fact, due to the nodes' sleeping behavior, we must deal with a sparse topology where connectivity is not ensured at all times. In spite of this, however, good broadcast algorithms should be able to reach all nodes in the network within a single flood, including those nodes that are asleep. To achieve these goals, we adopt the *Fireworks* approach presented in [9]. Fireworks is a simple probabilistic protocol which does not require any overlay network to be set up in advance. If the forwarding probabilities are correctly configured, all nodes in the network are reached with high probability and with low overhead. In fact, Fireworks reduces the number of links over which messages are sent with respect to flooding [14] and gossiping [8]. The analytical properties of the Fireworks scheme are detailed in [9] together with performance comparisons with respect to flooding and gossiping. Our interest here is in implementing the approach in practice and integrating it with the forward data dissemination phase (sensor nodes $\rightarrow$ sink). The Fireworks protocol is an on-line scheme working as follows. The sink transmits to all its neighbors. Whenever a node receives a new broadcast message, it re-broadcasts it to all its neighbors with probability p, while with probability $1-p$ it sends it to c randomly selected neighbors. c is usually lower than or equal to 4 [9]. Next, we present through an example how this is implemented together with on-line neighbor estimation algorithms. Let γ_i be the actual number of neighbors of a given node i, where we include both active and sleeping nodes. Let $\tilde{\gamma}_i$ be the current estimate of γ_i at node i. Upon receiving a broadcast message, node i with probability $1-p$ decides to send the packet to c neighbors, where c is a parameter of the algorithm. If $c > \tilde{\gamma}_i$ the node sends the message to exactly $\tilde{\gamma}_i$ neighbors, while if $c \leq \tilde{\gamma}_i$ the node sends the packet to c of its neighbors, which are randomly picked. On the other hand, with probability p the node sends the broadcast message to all ($\tilde{\gamma}_i$) its neighbors. Note that this is achieved through a neighbor discovery phase, i.e., by exploiting either the EBT or the WINDOW counting protocols. In practice, node i starts a new neighbor estimation/discovery round, by sending the message to either the first c or $\tilde{\gamma}_i$ neighbors that it counts within range, depending on the outcome of the coin tossing and on the value of $\tilde{\gamma}_i$. The whole procedure may be executed in multiple estimation rounds until 1) c neighbors have received the interest packet or 2) node i has successfully transmitted the interest packet to $\tilde{\gamma}_i$ nodes. Note that in the above algorithm interest dissemination and neighbor estimation are implemented in parallel and as different tasks of the same protocol.

E. The IRIS System

The IRIS system arises from the integration of the algorithms discussed above. A diagram of its main functional blocks and their interrelations is plotted in Fig. 3. Data dissemination (sink $\rightarrow$ nodes) is achieved through the integrated MAC/routing scheme of Section II-C. According to this solution, the next hop towards the sink is elected by means of locally and dynamically calculated costs aimed at weighing performance indicators such as state of the queues, residual energies, link qualities (SNR) as well as the advancements towards the sink provided by the candidate relay nodes. Estimates of the number of neighbors (N

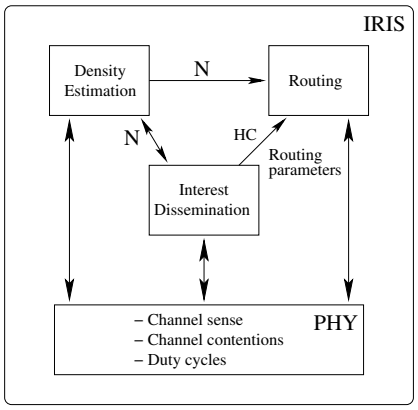

Fig. 3. A diagram of the IRIS system and its main components.

in the figure) are exploited by such a scheme to keep the latency experienced during channel contentions at a low value while minimizing the number of neighbors that unnecessarily contend for the channel. Estimates for the number of neighboring nodes are obtained through a density estimation algorithm that can be executed either alone or jointly with the interest dissemination (sink $\rightarrow$ nodes). The interest dissemination scheme exploits local density estimates (N) to check whether the forwarding rules of Section II-D are verified, i.e., when the packet has been successfully passed to *all* neighboring nodes (probability p). In addition, interests are used to refresh the hop count topology (HC) and send commands to the sensor nodes including: their duty cycle behavior, the cost function that shall be used for the calculation of the local costs in the data dissemination phase (routing parameters). This last feature allows for a very flexible data forwarding scheme, where different objectives such as data aggregation, energy and/or load balancing all translate into the exploitation of the proper cost function, without changing the channel access and routing mechanisms. All algorithms interact with each other and with the physical layer (PHY). Moreover, they can all be executed asynchronously by having them running concurrently in different portions of the network.

III. Simulation Results

In this section we present a selection of results from a thorough ns2-based [10] performance evaluation of our system.

Our simulations refer to scenarios where $n = 150$ to 300 sensor nodes are scattered randomly and uniformly in a square of side 200 m. The sink is placed at the center of the deployment area. Channel capacity is typical of sensor networking (38400 bps) and all nodes have a transmission range of 30 m. Two types of nodes are considered: Resource-rich nodes (*rich* nodes in the following), equipped with 240 Joules of initial energy, and *poor* nodes, with 48 Joules. The energy model used in our experiments follows the specifications of the sensor prototypes developed within the IST EYES project (`http://www.eyes.eu.org/`). Each experiment lasts 2000 seconds, which guarantees good statistical confidence. Sensors start generating data packets after an initial phase of interest dissemination initiated by the sink. Packets are generated uniformly according to a Poisson process with parameter $\lambda = 2$ packets/s per node. Data delivery to the sink happens according to the SARA protocol, where nodes are weighed by means of locally computed costs reflecting the percentage of consumed energy and queue occupancy. Hence, rich nodes are more likely elected as the next hops for data forwarding. The duty cycle period T has been set to 0.25 s and nodes act in a cooperative way (see Section II-A). The initial interest dissemination is performed by using the Fireworks-based interest dissemination protocol (see Section II-D) by considering $c = 4$, $p = 0.2$. All the presented results are obtained averaging over 100 experiments for each network size.

Fig. 2 and 4 show results concerning the estimation of the local density. In particular, we considered a scenario where 20 nodes are randomly distributed within the transmission range of an inquirer i. Both the EBT and WINDOW procedures have been implemented for counting i's neighbors. Fig. 2 shows the estimation error by varying the number of estimation rounds from 1 to 30. Both analytical (Eq. (1)) and simulation results are drawn when the duty cycle d is 0.1 and 0.5. Simulation points closely match analytical results. Interestingly, the estimation error obtained by simulating WINDOW is just slightly higher than what predicted analytically. This is mostly due to the collisions, which are not captured by the analytical model. In fact, due to colliding replies the inquirer underestimates the number of active neighbors (leading to higher estimation errors). However, this effect is substantially mitigated by the adaptive procedure adopted to set the window size. We also observe that a higher duty cycle corresponds to a lower estimation error, for a given number of rounds. In particular, the number of rounds needed to make the estimation error negligible is 8 for $d = 0.5$ (EBT), and around 45 for $d = 0.1$ (not shown in the figure).

When the neighbor estimation is integrated within our interest dissemination procedure the number of estimation rounds performed by each of the schemes investigated is in fact much lower. On average, less than 7 rounds are enough for a node that has to reach $c = 4$ neighbors to successfully do so. The same number of rounds are also enough for a node to complete the broadcast of the interest when a node has to transmit to all its neighbors. (In our experiments, the sensors reached a number of nodes equal to the total number of estimated neighbors within 7 rounds.) Note also that each neighbor counting round is reasonably fast (see Fig. 4). Even in dense networks ($n = 300$) little time, ranging from 0.6 s (WINDOW, $d = 0.1$) to 2.5 s (EBT, $d = 0.5$), is enough to complete the round. In addition, as shown in Fig. 4, the lower the duty cycle, the faster the single estimation round. This is motivated by the lower number of active nodes as the duty cycle decreases. The simpler the counting procedure (WINDOW vs. EBT), the lower the overhead per round (225 B vs. 870 B, $n = 300$ and $d = 0.5$), and the faster each round (1.3 s vs. 2.5 s, $n = 300$ and $d = 0.5$).

The time taken by the network-wide dissemination process to complete is reported in Fig. 5 ($c = 4$, $n = 200$). This metric accounts for the delays introduced by channel access, sleeping behaviors and neighbor estimation. As shown in this figure, the WINDOW scheme in many cases halves the delay with respect to EBT. Also, all schemes seem to be quite insensitive to the parameter p. This was expected and is due to the chosen value for c [9]. Finally, lower duty cycles lead to a longer dissemination time. This phenomenon is also expected as some extra-time is needed for the sleeping nodes to wake up and continue the interest dissemination process.

In our experiments we observed that the estimation error that can often be as high as 55% (WINDOW, $n = 300$, $d = 0.5$)

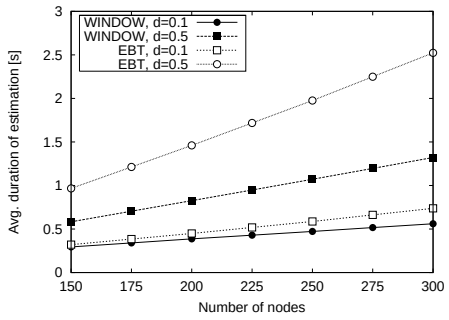

Fig. 4. Average duration of the estimation phase

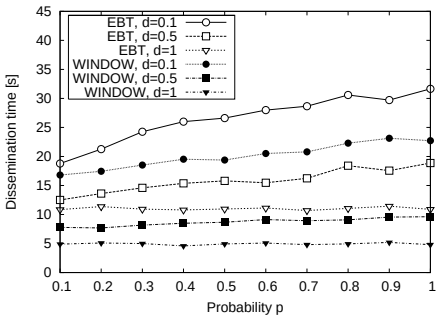

Fig. 5. Time taken for the network-wide interest dissemination to complete

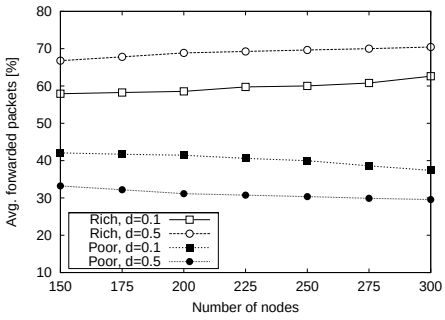

Fig. 6. Percentage of packets forwarded by rich and poor nodes

and as high as 47% in the case of EBT in dense networks. However, we observe that Fireworks-like interest dissemination is robust to small variations of the number of recipients of the interest packet (at each hop), resulting in a reliable interest dissemination process even in the presence of such estimation errors. In case of sparse networks ($n = 150$) 97% of the nodes receive the interest packet. This percentage grows to 100% when $n > 200$. WINDOW and EBT perform similarly with respect to this metric. Being faster, simpler and imposing lower overhead without significant loss in reliability, WINDOW proves to be an effective technique for being used in practice.

The results that follow refer to the performance of SARA. All the packets sent to the sink are successfully delivered. Sensor-to-sink packet latency is duty cycle-dependent. When $d = 0.1$ a node has only an average of 2 active neighbors closer to the sink ($n = 150$). This makes finding a relay quite a challenging task resulting in the need for iterating the relay searching process multiple times (e.g., 3 times in sparse networks), which implies higher latency. When $n = 300$ it takes an average of 1.5 s (5.1 s) to deliver a packet to the sink when $d = 0.5$ ($d = 0.1$). The average time to deliver a packet to the next hop is 0.15 s. The effectiveness of SARA in choosing as relays rich nodes more often than poor ones is shown in Fig. 6. At high network densities and for high duty cycles, each node can select among a large number of neighbors. This allows SARA to exploit at best the cost-based relay selection. When $n = 300$ and $d = 0.5$, 70% of the forwarded packets are handled by rich nodes, leading to effective energy balancing. In sparser scenarios and low duty cycles ($n = 150$ and $d = 0.1$) the degree of freedom of each node in selecting relays is lower. It is not always possible to select the best relay as sometimes the only available ones are poor. This imposes a reduction (to 58%) of the percentage of packets handled by rich nodes. We observe that when no rich nodes which are closer to the sink are available, a node prefers to forward the packet to a rich node at the same HC distance. This happens from 38 to 42% of the times. Higher duty cycles make it more likely to find rich nodes closer to the sink, thus resulting in a slightly lower percentage of times a node with the same HC is selected as relay.

IV. Conclusions

The paper presented IRIS, an Integrated Routing and Interest dissemination System for wireless sensor networks. The various protocols and functional blocks composing this system were first presented separately, and the advantages offered by their combination into an unique framework were discussed. In the final part of the paper, we reported performance results obtained through ns2 simulation of the whole system: overall, IRIS proved to be a promising approach for reliable and energy efficient operations in wireless sensor networks. Future work includes additional performance investigations as well as the implementation and testing of our system in a testbed.

References

[1] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 2–16, February 2003.

[2] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks," *ACM/IEEE Transactions on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.

[3] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Multihop Performance," *IEEE Transactions on Mobile Computing*, vol. 2, pp. 337–348, Oct-Dec 2003.

[4] H. Füßler, J. Widmer, M. Ksemann, M. Mauve, and H. Hartenstein, "Contention-based forwarding for mobile ad-hoc networks," *Elsevier's Ad Hoc Networks*, vol. 1, no. 4, pp. 351–369, Nov. 2003.

[5] Y. C. Tay, K. Jamieson, and H. Balakrishnan, "Collision-Minimizing CSMA and its Applications to Wireless Sensor Networks," *IEEE J. Select. Areas Commun.*, vol. 22, no. 6, pp. 1048–1057, Aug. 2004.

[6] D. Chen, J. Deng, and P. K. Varshney, "A state-free data delivery protocol for multihop wireless sensor networks," in *IEEE WCNC*, New Orleans, Louisiana, US, Mar. 2005.

[7] P. Popovski, F. H. Fitzek, and R. Prasad, "Batch Conflict Resolution Algorithm with Progressively Accurate Multiplicity Estimation," in *ACM DIAL M-POMC 2004*, Philadelphia, PA, US, Oct. 2004.

[8] Z. Haas, J. Halpern, and L. Li, "Gossip based ad hoc routing," in *IEEE INFOCOM*, New York, NY, US, June 2002.

[9] L. Orecchia, A. Panconesi, C. Petrioli, and A. Vitaletti, "Localized techniques for broadcasting in wireless sensor networks," in *ACM DIAL M-POMC 2004*, Philadelphia, PA, US, Oct. 2004, also to appear in *Algorithmica* as: D. Dubashi, O. Häggström, L. Orecchia, A. Panconesi, C. Petrioli, A. Vitaletti, "Localized Techniques for Broadcasting in Wireless Sensor Networks".

[10] The VINT Project, *The ns Manual*. http://www.isi.edu/nsnam/ns/, 2002.

[11] M. Rossi, M. Zorzi, and R. R. Rao, "Cost Efficient Routing Strategies over Virtual Topologies for Wireless Sensor Networks," in *IEEE Globecom 2005*, St. Louis, MO, US, Nov./Dec. 2005, also to appear in the *ACM Journal of Wireless Networks* as: M. Rossi, M. Zorzi, R. R. Rao, "Statistically assisted routing algorithms (SARA) for hop count based forwarding in wireless sensor networks".

[12] F. Ye, A. Chen, S. Lu, and L. Zhang, "A Scalable Solution to Minimum Cost Forwarding in Large Sensor Networks," in *IEEE ICCCN*, Scottsdale, AZ, US, Oct. 2001, pp. 304–309.

[13] M. Rossi and M. Zorzi, "Probabilistic Algorithms for Cost-based Integrated MAC and Routing in Wireless Sensor Networks," in *SENMETRICS*, San Diego, CA, US, July 2005, also to appear in the *IEEE Transactions on Mobile Computing* as: M. Rossi, M. Zorzi, "Integrated Cost-Based MAC and Routing Techniques for Hop Count Forwarding in Wireless Sensor Networks".

[14] Y. Yi, M. Gerla, and T. Kwon, "Efficient flooding in ad hoc networks: a comparative performance study," in *IEEE ICC*, Anchorage, Alaska, May 2003.