Simulation of Multihop Energy-Aware Routing Protocols in Wireless Sensor Networks

Adrian Fr. Kacsó
Computer Science Department
University of Siegen
57068 Siegen, Germany
Email: adrian.kacso@uni-siegen.de

***Abstract*—This paper provides and evaluates many simulation results concerning the energy consumption in WSN under different assumptions for various scenarios, including the impact of the routing strategy, broadcast delay, data aggregation, data rate, and the significant effect of MAC selection and configuration of its parameters. For simulations we analyze first the node's behavior in terms of energy consumption and investigate the impact of different parameters on it.**

To that aim, we use our sensor network framework (SNF), a flexible tool to build various protocols by combining existing building blocks at different layers (i.e., we provide also complete energy-efficient MAC modules). In this simulation environment routing protocols can be rapidly developed, closely inspected and the effects of changing configuration parameters and their impact on the performance better investigated and analyzed. We illustrate this in case of a two phase adaptive energy-aware routing protocol (with several routing metrics) as well as an enhanced, energy-aware directed diffusion and provide here various experimental results. After simulation and evaluation we are able to give guidelines for suited routing metrics and strategies, composition of protocols at different layers and how joint optimizations with MAC protocols increase the efficiency of routing.

***Index Terms*—wireless sensor network (WSN); routing protocols; energy-aware; simulation framework; modeling**

I. Introduction

A wireless sensor network (WSN) consists of a large number (hundreds, thousands) of sensor nodes that are randomly and densely deployed in a geographical area. Each of the distributed nodes in the WSN is able to collect large amounts of information, analyze and/or preprocess them and communicate them to a base node (sink). The nodes operate unattended and are forced to self-organize themselves as a result of frequent topology changes and to adjust their behavior to current network conditions. Typically, a sensor node has restricted communication (radio range) and computation capabilities, limited energy and memory. The communication is unreliable, messages can be lost or corrupted and sensor nodes can be damaged. The network topology changes also due to node transient failures, addition or depletion.

A very challenging aspect in query-driven WSNs is to determine the way the messages (query and data) are forwarded between the sink and sources (nodes able to deliver the requested data) using data-centric approaches. In such *data-centric routing* schemes the destination node of messages is specified by tuples of attribute-value pairs of the data carried inside the packets and not using globally unique identifiers (node address). WSN applications are usually interested in the kind of data and it is less important which node sent the data. When the distance between source(s) and sink is large, intermediate nodes forward the messages from hop to hop until they reach the intended destination, leading to several possible multihop paths. Determining which set of intermediate nodes to select in order to establish a path with the aim to prolong the network lifetime (by conserving the energy of the nodes as long as possible) is not trivial. Besides the energy-efficiency requirement a routing protocol for large WSNs must be reliable and scalable.

The paper is an extension of [1] and is structured as follows. Section II presents the state-of-art and the motivation behind designing energy aware routing protocols for WSNs. Section III describes the node software communication architecture. Section IV discusses how diverse routing protocols are built using our framework. Section V illustrates the performance of these protocols by giving various simulation results.

II. Related work, motivation and objectives

For WSNs, where multiple source nodes send data to a sink node (many-to-one communication pattern), establishing reverse paths is a very used scheme [2][3][4][5]. Many of the algorithms use distance-based forwarding, where the number of hops serves as a distance metric. Here each node selects the neighbor with the lowest hop counter to forward the packet. Since in most WSNs applications the battery of a sensor node is not replaceable, an important objective for routing protocols is the energy-efficiency. The biggest energy drain results from transmission of packets. Shortest-path routing improves the overall energy consumption since the energy needed to transmit a packet from source to final destination is correlated to the path length. Unfortunately, algorithms which minimize the path length will heavily load nodes on the path and those nodes drain off sooner, thus creating holes in the network, or worse, lead to disconnected networks.

Techniques to balance the load among all forwarding nodes are thus required [6][7][8][5]. One approach is to choose routes such that the variance in battery levels between different routes is reduced. By taking the amount of node's remaining energy into account we prevent nodes from choosing the same route often and thus increase the lifetime of frequently used

nodes on common used routes [9][6][10]. Minimizing the variance of the remaining energy of all nodes in the network is used in the lifetime prediction routing protocol [8], where the network lifetime is maximized. The lifetime of a node can be predicted based on the residual battery capacity and the rate of energy discharge.

In [7], Nurull et al. propose a route selection method that considers both the routing cost and the network lifetime metrics, achieving in this way a good tradeoff between these conflicting goals. Using a least cost route in order to optimize some cost (such as hop count, energy, delay, link quality, etc.) impacts on the network lifetime since nodes with higher communication demands might die soon.

According to Aslam et al.[9], the network lifetime maximization problem can be viewed as a max-min optimization problem. The proposed max-min zP_{min} algorithm selects routes that achieve a balance between the energy consumed by a route and the minimum residual energy at the nodes along the selected route. The basic idea is to select a route that uses at most $z \cdot P_{min}$ energy, where P_{min} is the energy required by the minimal energy route, and z is an adjustable parameter ($z \geq 1$). Between the routes with the total power consumption per path below $z \cdot P_{min}$, the route with the maximal minimum residual energy is selected.

Similarly, GBR [6] improves Directed Diffusion [2] by uniformly balancing the traffic inside the network using traffic spreading and in-network processing (aggregation).

Moreover, energy efficiency can be achieved by using greedy forwarding schemes which are aware of the geographic coordinates of the nodes as in [11][12]. For energy-saving approaches at MAC layer the reader is referred to [13].

Besides the energy constraint, a major concern in the design of WSN protocols is a reliable delivery of the data packets to sink. Radio communication links are known to have transitional region with widely varying degrees of packet loss and high error rates [5][14]. Although the successful packet reception decreases with the distance, there might be cases where distant nodes may have smaller loss than nearby nodes. That means that establishing energy-aware reverse paths using hop counter metrics might not always keep the overall packet transmissions energy to a minimum. In terms of energy, it might be more efficient to establish longer paths exhibiting low loss instead of shorter ones with poor link qualities where more energy has to be spent for successful packet delivery. In such cases new metrics are required such as the ratio of delivery rate and energy costs as in [5] or metrics that incorporate packet delivery rate and link asymmetry as in [14].

Besides communication links failures, sensor nodes can be damaged for a shorter or longer period of time. Routing protocols must tolerate such unreliable links and node failures. In the latter case, the routing protocol must react quickly to topology changes, especially when a route is affected by the failure of a node. The robustness to different types of failures can be improved by multipath routing [15][3][4][5], where multiple paths between source(s) and sink are established. Different strategies to construct disjoint and partially disjointed (meshed) paths and the tradeoff between energy-robustness are discussed in [15]. In Gradient Broadcast (GRAB) [4] and Minimum Cost Forwarding [3] packets travel to sink by descending a cost path. The cost is defined as the minimum energy needed to forward packets to the sink along previously established routes. All nodes receiving a packet with smaller cost forward it, meaning that the packet can reach the destination along several routes. This improves the reliability but increases the energy consumption since the packet is transmitted (superfluously) on more paths. To improve the energy consumption one can use, for example, multi-link energy-efficient forwarding as in [5]. In contrast to single-link forwarding, where the sender sends packets to one forwarder, the multi-link forwarding exploits the broadcast characteristics of the wireless shared channel. The idea is to broadcast the packet to a predetermined potential forwarder set. If the first node in the ordered forwarding set does not acknowledge the packet, the sender polls the next node in the set.
Note that reliability needs to be implemented at upper layers (as in Directed Diffusion [2]) when the transmission does not employ a MAC layer reliability mechanism such as the RTS/CTS/ACK handshake [16].

The behavior of the nodes between source(s) and sink(s) depends on several factors such as the network topology and connectivity, the number of active requests in the network, the position of source(s) and sink(s), communication pattern, MAC protocol parameters (e.g., listen and sleep times), application parameters (e.g., interest refresh rate) and so on. The cumulated impact of such a large amount of parameters on the (routing) behavior of individual nodes is difficult to be predicted. Our main goal is to simulate reliable routing protocols for WSNs able to prolong the network lifetime by conserving the energy of the nodes as long as possible. Therefore, we look after new metrics based on a sensor node's residual energy or other attributes to be able to take appropriate routing decisions in order to extend the lifetime of the WSN. Moreover, the energy consumption must be optimized at each layer of the node communication architecture and the cooperation between layers should be improved. We proposed in [17] a modular, energy-aware network architecture of a sensor node as a flexible approach to design and plug-and-play various protocols at network and MAC layers, and to combine and analyze the impact of different strategies (inside the protocols) on the performance and lifetime of the WSN. We implemented the SNF simulator using the OMNeT++ 3.4b2 discrete event simulation package[1] [18] and its Mobility Framework (MF[2]) 2.0p3 [19]. Part of the MAC protocols and the automatic energy component of the framework were described in [20]. Moreover, we extend the framework with new features such as add/delete, move (drag-and-drop), disconnect/reconnect nodes

[1]OMNeT++ is a public-source, component-based, modular and open-architecture simulation environment with applicability in the simulation of communication networks (http://www.omnetpp.org/)

[2]extension intended to support wireless and mobile simulations within OMNeT++

during simulation, which allow to analyze the impact of dynamic topology changes.

In the present paper we focus on alternatives to design energy-aware routing protocols using metrics that prolong the network lifetime and we illustrate the performance of these distributed algorithms using our automatic evaluation tool (based on our statistic component). The quantitative effort to design new protocols and to integrate them into a complete protocol architecture is considerably reduced since we promote modularity (in design) and code reusability. We illustrate here this by implementing two different routing protocols.

III. Node architecture

The common approach used to structure a communication protocol is layering; separate protocols are building the protocol stack where each protocol accesses functions of the lower layer protocol. Since for a resource constrained node strict layering is inappropriate [21][22], we employ a cross-layer design [23] by allowing exchange of information mainly across application, routing and MAC layers in order to optimize them. For routing protocols such optimization may improve the energy consumption during communication, extend the spectrum of routing decisions and adapt the communication to tolerate different kinds of failures or to avoid local congestion.

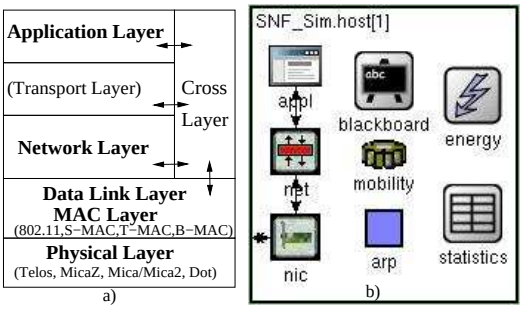

Fig. 1. a) Components of a sensor node b) in simulator.

The software comunication architecture of a sensor node is illustrated in Figure 1 and consists of application, network and NIC layer; the latter incorporates the MAC layer, the physical layer and the radio. We provided full implementations for the application and MAC layers, we extended the mobility component and added energy and statistics modules (see [20]). The existence of the blackboard component (module) allows cross-layer interactions between the layers.

Recall that we are looking for energy-aware routing protocols for long term query driven applications under the assumptions that the message sequence is not known in advance and several sinks inject requests in a large random sensor network.

A. Application layer

The application layer provides the user a general way to send his request to the network. In a data-centric approach, a request or *interest* is a sequence of attribute-value pairs such as *type=temperature, interval=100ms, area=[(x,y),(u,v)]* which is converted to an application packet. A request can be sent to a given area if the user specifies a rectangular area (defined by the coordinates of the two points). The application layer contains and simulates the sensing unit of a sensor node and sends responses or *data events*. Upon receiving a request, the node checks if it is a source of the requested data; if yes, the application layer starts the requested sensor to gather the data and packs it in a response message to be sent to the request initiator (sink). To that end, we employed one application layer than can be used by all routing protocols.

B. MAC layer

At MAC layer, the main energy consumer in a sensor node is the transceiver. To accommodate a low energy consumption, the main idea is to turn off the transceiver most of the time and to activate it only when necessary, meaning that it works at a low duty cycle. Thus, we provide implementations (as complete NIC modules) for energy-aware MAC protocols like S-MAC [24], T-MAC [25], Preamble Sampling [26] and IEEE 802.11 WLAN standard that can be used below a network layer routing protocol (including support for collision detection). To the best of our knowledge, so far there are no implementations for different MAC protocols (except a first attempt of the IEEE 802.11 WLAN) which can be embedded in the OMNeT++ sensor node architecture. This contribution will be made publicly available for download and is not the focus of this paper.

C. Network layer

In order to quickly build and experiment with different routing protocols, the network layer should be designed to allow fast reconfiguration and code reusability.

At network layer the possibilities to reduce the energy consumption are to communicate rarely and to reduce the volume of communication, i.e., the number of transmitted packets by using in-network processing (aggregation, compression, etc). In WSNs the network layer provides a (best-effort) connectionless multihop communication abstraction to the upper layers. Typically, its functionality includes packet forwarding towards one or many destinations, creation and maintenance of routing structures, retransmissions and acknowledgments. Note that the general functionality of a routing protocol remains the same, what is changing from one protocol to the other are the protocol logic and the strategies which use different metrics to route the packets. In order to better exploit this, we implemented in SNF the network layer architecture proposed in [17]. We shortly describe here the architecture (Figure 2), since we will have to refer to most of its components and their interactions.

It consists of three components: the in-out unit (IOU), the forwarding unit (FU) and several routing units (RUs). The main task of IOU is to guide the packet flow. The FU forwards the packets to the appropriate RU according to the packet type and the direction they are coming from. It also maintains and provides access to internal cache structures (Interest, Gradient and Neighbor Tables, etc.). The *Interest Table* is a dynamical, user-programmable table containing request relevant information. Since we allow more sinks to inject concurrent requests

into the network the interest is uniquely identified by the attributes: a node identifier, the type of requested data and the area under observation. Each interest entry contains a reference to a dynamical, user-programmable *Gradient Table* which contains the direction (node id) the packet came from and other routing relevant information. Unlike the Gradient Table, the *Neighbor Table* contains neighborhood information that is independent of the interest (energy, link quality or times when neighbors start and end their active period according to the used MAC protocol, etc.).

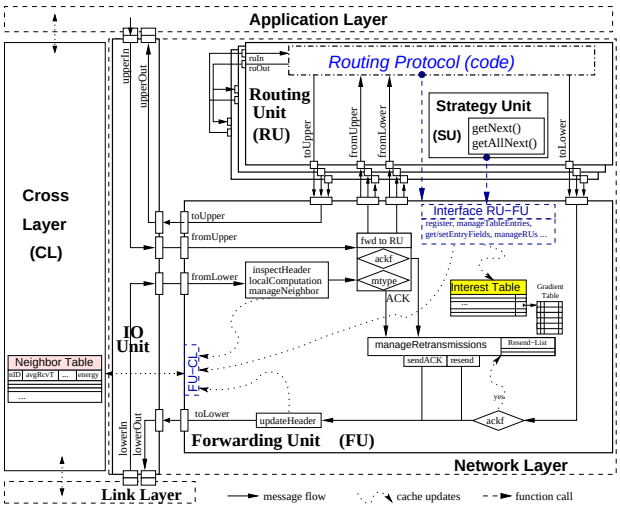

Fig. 2. Interaction of the components at network layer.

The RU is a dynamically exchangeable component implementing (part of) the routing protocol with the main task to forward packets by determining their next hops. The SU is an optional component inside the RU with the aim to modularly separate the policy of determining the next hops for packets inside the routing protocol. Since the architecture is not subject of the presented paper the reader is referred to [17] for more details.

D. Other components

Furthermore, we integrated in the node architecture:

- an energy component that models automatically the energy consumption in a sensor node (including also the energy consumption due to collisions, in order to have a realistic energy model),
- a statistics component with automatic visualization of relevant data, which enables a fast analysis of different performance criteria,
- a cross layer enabling interchange of significant information directly among the different layers of the protocol stack (without additional messages).

IV. Energy-aware multihop routing

Here we concentrate on two routing protocols that we implemented at the network layer: a two phase multihop routing and enhanced directed diffusion.

A. Two phase multihop routing

This is an adaptive energy-aware routing protocol based on two phases: the *interest propagation* and the *data transmission*, where the data is sent along the reverse paths established during propagation of the interest. The main idea of this routing is to find out what is the relevant routing information that shoud be spread to the nodes in the network without sending explicit routing messages.

During the first phase the interest packet propagates throughout the network, the cost field (hop count or other metrics) is established at each node and the gradient tables are created and initialized. Finally, each node has determined its minimal cost to sink and depending on the size of its gradient table, it knows a subset or all of its neighbors and their energy.

In the *data transmission* phase, the data packets are routed from source(s) to sink according to the minimum cost forwarding principle. The choice of the next hop is based solely on information available inside the gradient table and therefore the corresponding routing algorithm is sender initiated, where each involved node selects the ”best” next hop and sends data as unicast. This best next hop can be chosen according to several strategies (metrics).

The first strategy was to route the data packets on the shortest path between source and sink (hop count metrics, denoted in the sequel `hc`).

The second strategy, denoted `hc,E`, combines the hop count metrics with the neighbor residual energy. More precisely, each node records the neighbor with smaller hop count than itself and among these the node with the maximal residual energy is selected as its best next hop. However, choosing the neighbor with the highest energy level does not guarantee that the path to the sink along this node contains only relay nodes with high residual energy. It can occur that this path contains a bottleneck energy node, which should be avoided whenever possible.

In order to overcome this we consider a third strategy, denoted `Δhc/E`, which combines the hop count and the residual energy of each node on the entire path between source and sink. To that end each node u computes its cost as $M_u = \min\{M_v + 1/E_u | v \in Neighbor(u)\}$, where E_u is the residual energy of node u and M_v is the summation of the costs on the path from the sink up to and including node v.

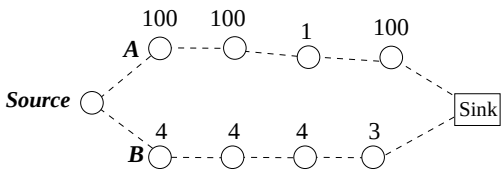

Fig. 3. Two disjoint paths from source to sink. The residual energy for each node is given.

This additive path cost function represents now a quantitative characterization for the goodness of the entire route. For example, if a relay has three alternatives nodes A, B and C with costs $1/20$, $1/10$ and $1/5$ respectively, it will choose the

route going through node A, since this obeys the minimum-cost forwarding principle. This third metrics contributes to better balance the energy consumption of the network by redistributing the traffic load more uniformly on the nodes. Still, there are particular scenarios where this strategy does not avoid a bottleneck node, as illustrated for a source with two disjoint long paths in Figure 3. The cost on the path through A is $1/100 + 1/100 + 1/1 + 1/100 = 1.03$ wheras through B is $1/4 + 1/4 + 1/4 + 1/3 = 1.083$. Thus Δhc/E selects the path with the smaller cost through A although it contains a bottleneck node with residual energy 1 (which will be soon depleted).

Alternatively, in order to avoid such node with low residual energy we introduce a forth strategy, denoted hc,cE, which employs uncorrelated the hop count and the critical energy (cE) on the entire path between the sink and source (i.e., the least residual energy on this path). Each node u computes $cE_u = \min\{E_u, \max\{cE_v | v \in Neighbor(u) \ \wedge \ hc_v \leq hc_u\}\}$, hence the hop count plays the main role in selecting the next hop, while the critical energy on the path just refines the decision. In order to enable this, each node maintains in its cache tables (§III-C) information about its neighbors including hop count, critical energy and the timestamp of last received critical energy message. To forward a data packet to sink a node uses the hop count and critical energy metrics (where the weight of each factor is adjustable). The node selects its next hop candidate (*cand*), out of three sets: nodes with smaller (*Set1*), equal (*Set2*) and higher (*Set3*) hop count (*hc*) by executing the pseudocode:

```
cand = cand1
if (cand2 ∈ Set2)
   if ((cand2.cE - cand.cE) ≥ eDiff1)
      cand = cand2
if (cand3 ∈ Set3)
   if ((cand3.cE - cand.cE) ≥ eDiff1*(1+(cand3.hc-cand.hc-1)*k)
      cand = cand3
```

eDiff1 and *k* are configurable threshold values. Note that for *Set3* we relax the condition $hc_v \leq hc_u$ to enable selecting neighbors that are one hop further away from the sink than the current node. The drawback here is the possibility of creating loops in the routing process since we lose the "right direction" information kept inside the hop count.

In order to avoid the drawbacks of strategies three and four we propose a further new strategy, referred as hccE, which correlates both the hop count and the critical energy on the path. Therefore, each node computes and forwards the pair: [hop count distance to sink; critical energy on path], as $(hc_u; cE_u) = (hc_v; cE_v) \oplus (1; E_u)$, where $(hc_v; cE_v)$ is the hop count and critical energy pair corresponding to node $v = arg\min\{hc_v/cE_v | v \in Neighbor(u)\}$ and the operator $\oplus$ is defined for each term as $hc_u = hc_v + 1$ and $cE_u = \min\{cE_v, E_u\}$.

Additionally, to alleviate the problem of excessive broadcasts during flooding caused by the fact that a node broadcasts instantly after receiving a lower cost without knowing whether this cost is minimal we introduce a waiting time T_w. A node will wait for a time T_w which is chosen either constant or directly proportional with the received cost field. During this period, the node extracts from all received packets the minimal cost field and if this is better than its own, it updates its local cost. Then it broadcasts the packet with its cost. It is obvious that if T_w is large enough the node broadcasts only once the minimal cost, but this introduces latency in the transmission.

To guarantee reliable delivery (per hop), data packets are unicasted using RTS/CTS/ACK handshake at MAC layer or they are marked as relevant at network layer (which enables the network layer reliability mechanism).

Note that the interest is refreshed (broadcasted) at configurable intervals depending on the data generation interval. The refreshes are necessary in order to notice changes in the topology (new or failed nodes) and to propagate the path critical energy information in the network.

B. Enhanced directed diffusion

To illustrate the usefulness and the possibilities of the proposed SNF we present here also how a complex routing protocol such as directed diffusion (DD) can be realized. The protocol we implemented, referred as enhanced, energy-aware directed diffusion (EDD), is a variant of the original DD [2], including energy-awareness mechanisms (metrics which consider the residual energy of nodes and geographic coordinates [11]). The directed diffusion protocol is considered complex since it combines discovery, querying and routing mechanisms in a single protocol. Therefore, we decided to decompose it in phases, which can be modularly embedded in our architecture of the network layer. We discuss how we decomposed the protocol in phases (including our changes to the original) and show how its complexity can be reduced by employing simple routing units, which later can be exchanged and variably configured (by reusing the code) to achieve an energy-aware routing.

We briefly describe the four phases and the different types of messages used; we name the phases according to the operations the sensor node executes in each of them. Generally, the operations correspond to some known traffic patterns (given below for each phase in parenthesis) in the WSN:

Phase 1: Interest propagation (flooding, 1:all)
A sink aiming to subscribe to certain events creates an interest and injects it in the sensor network. The original interest has a low data rate and is broadcasted to the one hop neighbors (Figure 4.a). Intermediate nodes receiving the interest create and add a gradient entry in their gradient table, containing the interest attributes and the sender node from where the interest was received. These gradients allow the node to route back data matching the interest (a node knows only the one hop neighbor which sent the interest, not also the initiator of the interest). If the interest has not yet been seen, it is rebroadcasted by each intermediate node. This way the message is forwarded hop by hop in the entire network.

Phase 2: Path establishment (convergecast, k:1)
Once the interest reaches potential sources, the sources reply with an exploratory data message at the (low)

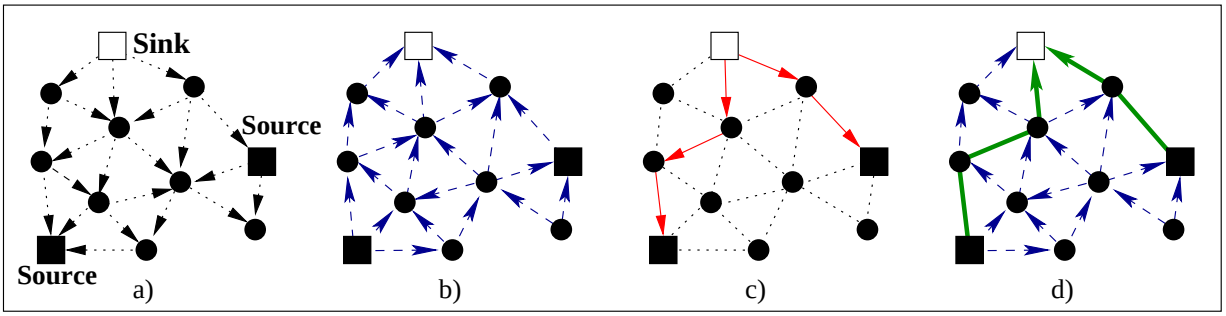

Fig. 4. EDD phases a) interest propagation, b) exploratory data convergecast, c) reinforcement of one neighbor to each source d) data propagation

requested rate, in order to find paths to the sink. Using the gradients (corresponding to this interest) from phase 1, the exploratory data messages are sent back hop by hop (via multiple paths) to the sink (Figure 4.b). This phase is referred in original diffusion as initial gradient setup phase.

Phase 3: Reinforcement (multicast, 1:k)

The sink receives one or more exploratory data messages. It selects the *best*[3] exploratory data message and a reinforcement message is sent towards the neighbor sending this message. Upon receiving a reinforcement message, each node creates a reinforced gradient for that neighbor, chooses the best next neighbor towards the source and resends the reinforcement message to the selected neighbor (Figure 4.c). Recall that the selection process (i.e., which node should be reinforced) is a local decision based on the contents of the data cache. This data cache (in our architecture the `DataInitiator Table`) is similar to the gradient cache but in the opposite direction, to route messages (reinforcement) from sink to source nodes. To be able to reinforce a given neighbor each sensor node stores, for each known interest, a data cache with recently received data messages. If the same data message is received several times, it is silently discarded. Thus, an intermediate node knows only two things: where to forward the incoming data message and which neighbor has been reinforced.

Phase 4: Data delivery (restricted convergecast)

When the reinforcement message reaches a source, a complete path of reinforced gradients exists between source and sink (Figure 4.d). Data messages can now be routed from each source to the sink using exactly one path.

In each phase a special message type is forwarded inside the network. There are several relevant messages types:

- **Interest** – Each time a request is injected at the sink an interest message is created. The interest is a set of attribute-value pairs containing at least the type of the requested data, a data generation interval and the expiration deadline for the interest. The original interest message is flooded, each intermediate node maintaining a gradient entry towards the sender of the interest. The sink refreshes the interest message periodically, in order
  - to announce that it still wants the data,
  - to update the node state and to discover topology changes (i.e., new/depleted sensor nodes),
  - to reach all nodes in case one or more of the previous interest messages were lost (due to collisions, communication failures, etc.).

  Additionally, if the network has been partitioned, the absence of the refresh message notifies the source that the sink cannot reach it. Correspondingly, the source may decide to delete the request after a given time. Between the original interest and the refresh interest there is no difference.
- **Exploratory data** – The exploratory data message is used by the source to explore or discover a path to the sink, as the sink and the source nodes do not know each other. When a node receives an interest message, it inspects the request and, if the data information that it can deliver matches the requested data, it declares itself a potential source for that interest. Data messages are initiated by a source and are forwarded (with the identity of the source) to *all* gradients in order to reach the sink. This creates multiple exploratory data messages reaching the sink. These exploratory data messages are cached at each node; they are refreshed periodically (with a lower data rate) for the same reasons as the interest message.
- **Reinforcement** – The positive reinforcement message is a modified interest, usually with a higher data rate, sent by the sink upon receiving one or more copies of an exploratory data message. In order to reduce redundancy of data messages, the sink selects one or *several* preferred neighbors as starting points for path(s) to reach the source(s). The selection criterion is usually latency, that means the fastest[4] neighbor (the first node that sent a not yet known exploratory data message) is selected. Checking the initiator of these data messages is possible only by caching the exploratory data messages and being able to distinguish among them.

  The reinforcement message is forwarded by each in-

[3]The meaning of the term varies according to the goal of routing. For example, when low latency is required the fastest neighbor is selected.

[4]Other possible criteria are: energy, hop-count, etc.

termediate node, based on the same next hop selection criterion until it reaches the source(s). Additionally, each intermediate node upon reception of the reinforcement message marks the reinforced gradient.
The **negative** reinforcement message is a path maintenace message. If a new improved path is discovered, the node has the possibility to use the better path and to diminish the importance or to deactivate the previous one. The old path can be used as backup path or becomes a normal gradient.

- **Data** – The data messages contain the actual data. A source initiates with the interest's requested rate periodically a data message, which has to be forwarded along the reinforced path. Each intermediate node can perform in-network processing on the data message received. The most common processing is a store and forward function, where some kinds of data reduction (e.g. aggregation) or filtering are performed before forwarding it.

According to the identified phases we can now split the protocol in several routing units as illustrated in Figure 5.

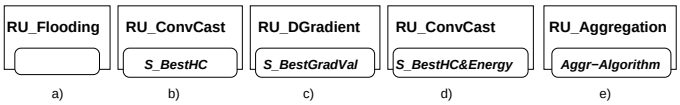

Fig. 5. a) Interest propagation, b) Convergecast using a subset of the gradients, c) Reinforcement using a subset of data-gradients, d) Data delivery and e) Aggregation.

We employ five routing units, the first four corresponding to the four phases of the protocol and the fifth one is an aggregation unit responsible to control the aggregation of data messages. Each of the routing units (excepting the `RU_Flooding` unit) has an attached strategy unit which specifies its policy (to achieve its goal).
The `RU_Flooding` unit, responsible for the interest propagation, does not need a strategy since it uses flooding (without any decision to be taken). The communication traffic cannot be reduced during this phase (except for geographic flooding, see §V-I), but the gathered information will be employed to considerably reduce the traffic in the subsequent phases.

Unlike the original directed diffusion where data messages are forwarded to all gradients (still flooding in phase 2), in our enhanced version we restrict the set of gradients to several of them according to a customizable criterion. We implement this phase in the `RU_ConvCast` routing unit and allow the use of any metrics (from simple one like hop count, latency, energy, etc. to combined ones like in §IV-A). Furthermore, the sources start to send their data exploratory messages only after receiving several refreshes for the same interest, in order to give the network time to stabilize.
The second and forth routing units are the same since both phases propagate data messages (either exploratory or real data messages). The only difference consists in the strategy (unit) used: in Figure 5 the `S_BestHC` strategy forwards the (exploratory) data messages in phase 2 (only) to neighbors with smaller hop count and the `S_BestHC&Energy` strategy to neighbors that besides a smaller hop count have also enough residual energy. It is also possible to consider for the second phase a strategy that consider principally the energy on the path without the hop count. In this way one get maybe longer paths and in the fourth phase one can use a hop count strategy to select the shortest path between them.

The third routing unit `RU_DGradient` uses data gradients (gathered in the second phase) and reinforces the nodes on the path according to the `S_BestGradVal` strategy. This strategy uses a path additive metrics (similar to the $\Delta hc/E$ strategy from §IV-A), where the path along nodes having better values are selected. Since the reinforcement process works always the same, but the decision which path to reinforce (the fastest one, the path with highest residual energy, etc.) the user can choose among all these (loadable) `SU`s the one more appropriate for his application.
Due to the fact that we generally keep several reinforced neighbors (according to the metrics) also in phase 4 we offer flexibility in chosing an energy-aware policy in order to better balance the data transmission load among the reinforced paths.

Hence, our enhanced version of directed diffusion allows a much better tuning during the protocol, which offers more flexibility and leads to a gradually and considerably diminuation of the communication traffic.

As each cost-field approach, directed diffusion scales well for large networks since the number of gradients kept in nodes depends on the number of requests and density (neighborhood) not on the number of sensor nodes.

V. Simulation results

The ultimate goal of running a simulation is to provide results and to get some insight into the behavior of the sensor network by analyzing the obtained results. We start with the two phase multihop routing protocol given in §IV-A using a 48 nodes WSN, continue in Section §V-I with a simulation scenario for EDD and conclude with some remarks about the reconfiguration of our simulator.

A. Impact of the routing strategy on the energy consumption

The application requirements assume that the data generation interval is set to $200ms$ and the request is refreshed at a $5s$ interval. We assume one sink (node 21) and a zone with only one source (node 16) as illustrated in Figure 8. Simulation results for the impact of different routing strategies on the energy consumption are given in Table I.

48 nodes net Energy consumed	**Strategy1** hc (hop count)	**Strategy3** Δhc/E (energy)	**Strategy4** hc,cE (critical energy)	**Strategy5** hccE combined
Max [mJ]	5.520	4.957	4.402	4.935
Max-Min[mJ]	3.625	3.075	*2.507*	*3.038*
Std. dev.[mJ]	*1.140*	977	*797*	*989*
Total [mJ]	*145.689*	147.972	150.557	148.769

TABLE I
Impact of strategies on the energy consumption.

As can be seen, an energy-aware approach leads to a better overall behavior of the network. More precisely, the results show that the better metrics are combinations of hop

count with critical energy (strategies 3,4 and 5); hereby the overall energy consumption is slightly larger (less than 3.3% in comparison with strategy 1), but the consumption is better balanced among nodes (up to 30% smaller standard deviation than for strategy 1), which extends the network lifetime.

When using only the hop count metrics we notice a very unbalanced energy consumption of the nodes (see standard deviation, the difference Max–Min); nodes on minimal hop count paths will soon be depleted while nodes with enough energy on comparable paths are not employed at all.

Figure 6 illustrates the *sorted* energy consumption of all nodes using T-MAC protocol with all the four strategies given before. The energy is sorted in order to better visualize the distribution of the energy consumption on the nodes. One can notice that strategies 4 and 5 using both the hop count and critical energy have the positive effect that they balance the overall energy consumption on more nodes. This can be observed by comparing the plots of *Tmac-hc,CE* and *Tmac-hcCE* with *Tmac-hc*; in the first two plots the curve is smoother than the last one, with no more than 3.3% overall energy increase (see Table I).

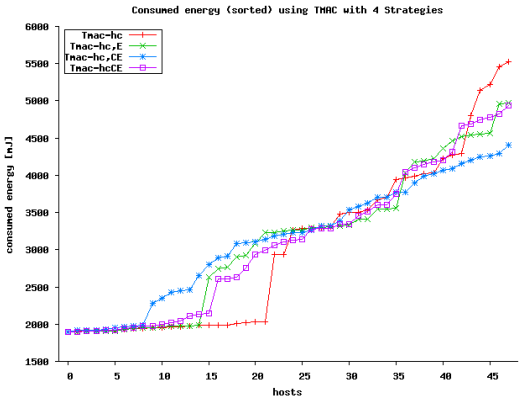

Fig. 6. Energy consumption using different strategies (same T-MAC).

To analyze the energy consumption on each individual node one can plot the energy unsorted as illustrated in Figure 7.

One can separate the nodes according to their position to the established path in three classes: a) nodes on the path (relay data messages according to the requested data interval), b) 1-hop neighbors to the path (receive data message updates and react protocol dependent on receptions for which they are not the intended receivers) and c) nodes more than 2-hops away from the path (not involved in data message transmissions, they receive interest refreshes or synchronization messages). The energy consumption decreases according to these classes. For example, for the first strategy using the hop count metrics, the nodes on the path, namely 1-7-2-11-20-21 in Figure 8, consume more energy than the other nodes.

B. Cumulative impact of the broadcast delay and strategy

We illustrate further the impact of the broadcast delay (T_w) on total energy consumption. It would be expected that for larger values of T_w the energy would decrease. This is true for MAC protocols without or with a fixed active-sleep regime.

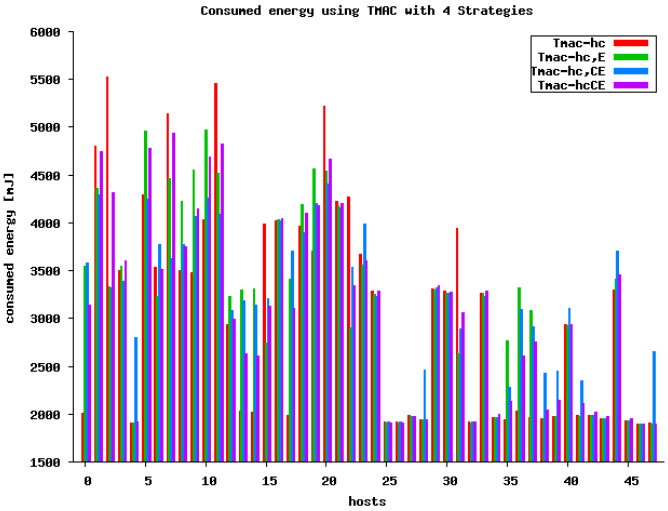

Fig. 7. Energy consumption of each node using the four strategies.

In case of low duty-cycle protocols, like T-MAC, the situation can be somewhat different. Adjusting T_w to achieve a better energy consumption remains difficult and we give some results for simulations configured with T-MAC. Nevertheless, the cumulative impact of different link (MAC, radio), network and application parameters should be further studied/analyzed to find out if an optimal value for T_w exists.

Since we set the listen time and frame time for T-MAC to 30ms and 600ms respectively, we run the simulation for three different values for T_w: 0.4ms, 20ms and 600ms. The simulation results for `hc` and `hccE` strategies are given in Table II, where the total energy consumption in WSN is given for each value of T_w. The simulation time was 2min and each of the 3 sources generates 5 data packets/s.

Energy [mJ]	Broadcast delay (T_w)		
Strategy	0,4ms	20ms	600ms
`hc`	85.403	84.759	85.221
`hccE`	91.868	89.964	90.964

TABLE II
IMPACT OF T_w ON ENERGY CONSUMPTION.

The best energy consumption is achieved with a broadcast delay of 20ms, thus not for the highest value of T_w.

We next check the impact of these values on the number of rebroadcasts.

Rebroadcasts	Broadcast delay (T_w)		
Strategy	0,4ms	20ms	600ms
`hc`	1.067 (2)	1.053 (4)	1.052 (4)
`hccE`	1.534 (2)	1.268 (0)	1.063 (1)

TABLE III
IMPACT OF T_w ON THE NUMBER OF REBROADCASTS.

In order to count how often each node rebroadcasts an interest we stop the refreshes after 112s (the simulation runs until 120s). In this way, we guarantee that each interest refresh reaches all the nodes (no one is still propagating). That means that each node should broadcast at least 22 times (interest

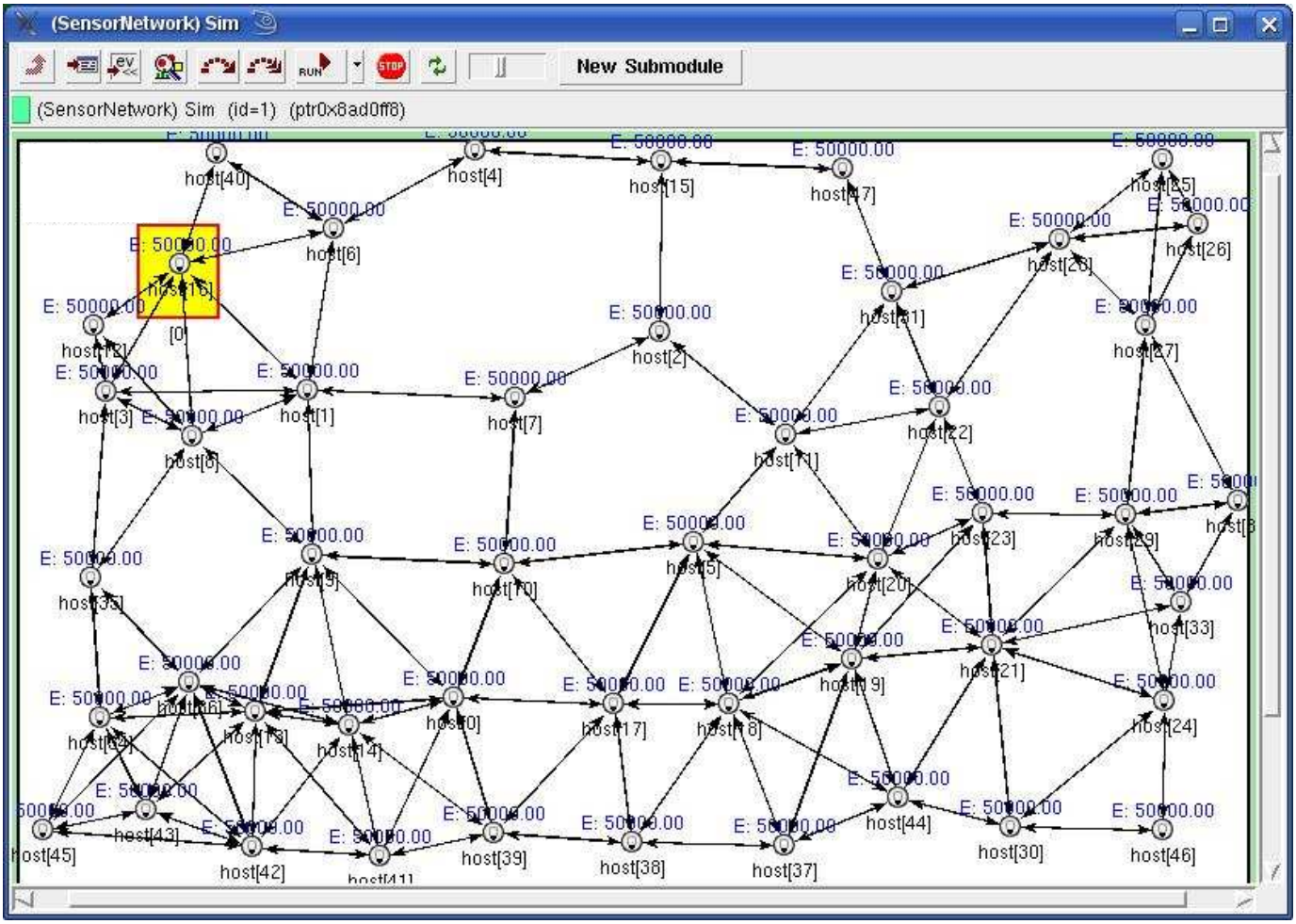

Fig. 8. The standard network with 48 nodes used in most simulations (to identify easily the nodes referred in the text; later snapshots are relatively small).

refresh rate is 5s), i.e., for 48 nodes this gives a total of 1056 times (optimum). The total number of rebroadcasts is given in Table III, with the number of missed refreshes in parenthesis (due to collision not all refreshes are received by all nodes).

From Table III one can observe that for the hccE the total number of rebroadcasts improves (near optimum) as T_w increases. For a broadcast delay of 0.4ms the number of rebroadcasts is high and therefore by using hccE strategy a T_w greather than 20ms is recommended. The distributions of rebroadcasts on each node for both the hc and hccE strategies are visualized in Figure 9 and 10, respectively.

Note that the value of T_w plays an important role for the hccE strategy, since here the metrics changes faster. Even though the number of rebroadcasts for T_w=600ms is near optimum, the energy consumption does not improve. This suggests that besides the number of rebroadcasts there are other factors that affect the energy consumption. We supposed that this can be caused by a higher number of collisions. Therefore, we measured the number of collisions and we found out that both strategies have slightly the same number of collisions for the same T_w (excepting hccE strategy with 0,4ms delay). That

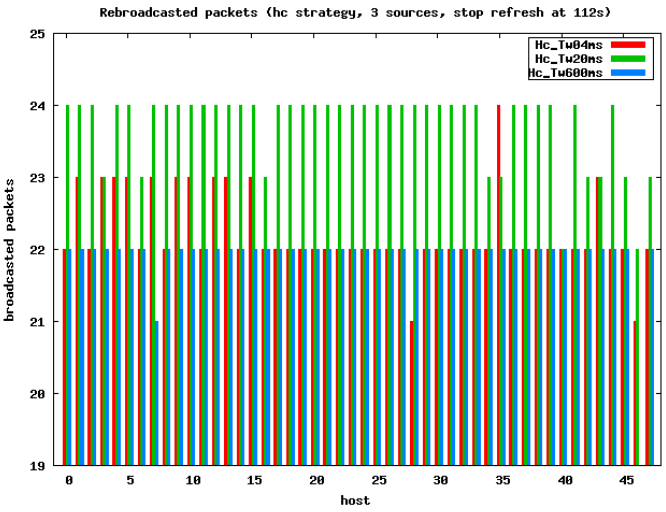

Fig. 9. Rebroadcasted interests for hc strategy.

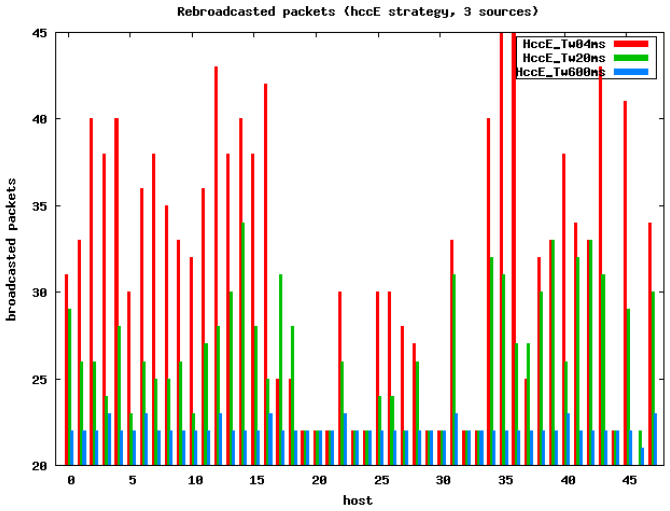

Fig. 10. Rebroadcasted interests for hccE strategy.

means that the better energy consumption for T_w=20ms is inherent to T-MAC's aggressive time-out policy. Since T-MAC extends its listen period at each send/receive event, the total time the node is in idle state is longer for a 600ms delay than for a 20ms delay.

C. Data aggregation

Since data readings are most of the time correlated, one can use in-network processing in order to reduce transmission. We consider now a simulation scenario with 3 sources (placed inside the rectangle in Figure 11), each one sending 500 packets at an interval of 200ms, with a simulation time of 2 min. By letting 3 sources to send simultaneously, the traffic load increases and each source tries to send its data packets on the shortest path (using hc strategy) or on an optimal path with the greatest critical energy (hccE strategy).

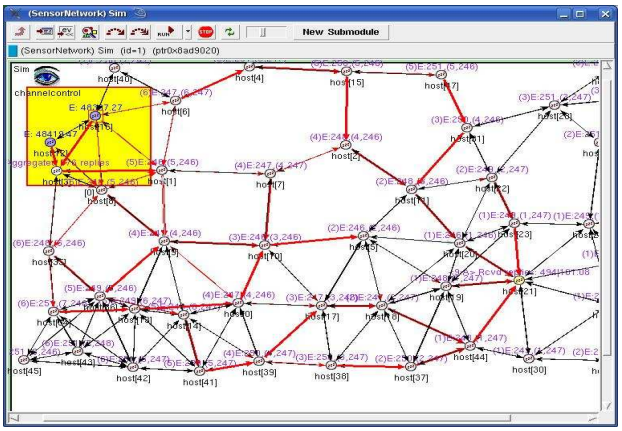

Fig. 11. Snapshot of the network with 48 nodes running strategy 5. The routes followed by the aggregated data messages are highlighted (red arrows).

For the hc and hccE strategies the energy consumption of nodes with and without aggregation (green and red curves, respectively) are given in figures 12 and 13. The energy gain is 32% for the hc strategy and 35% for the hccE strategy, respectively. One can notice that when aggregation is enabled not only nodes along the used paths consume less energy but also their neighbors.

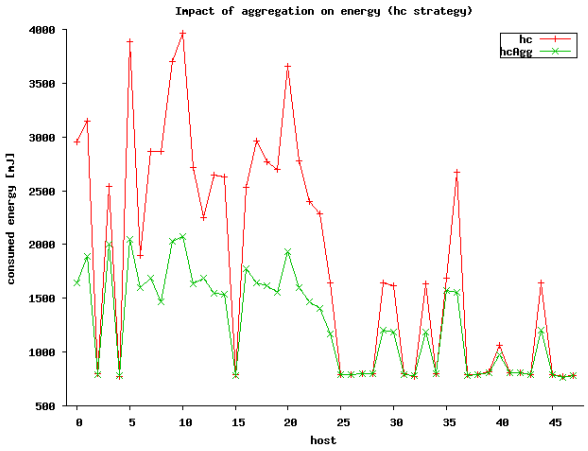

Fig. 12. hc: energy consumption of the nodes with/without aggregation.

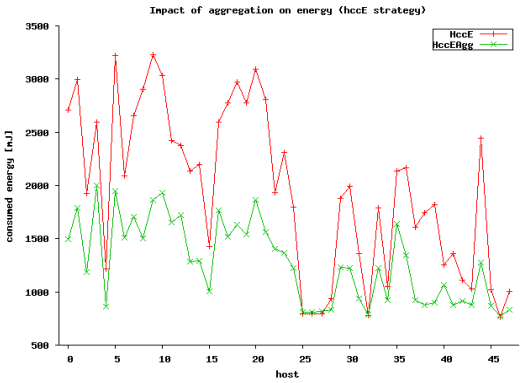

Fig. 13. hccE: energy consumption with/without aggregation.

The number of data packets sent by the sources and received by the sink reveals that no data packet (3 x 500) was lost on the way to sink (for place reasons we omit the result).

To aggregate the data messages the sources are building an aggregation tree (the aggregation algorithm is implemented by a different RU) inside the zone. The best positioned node becomes aggregator, it waits for data messages from the two sources and sends one aggregated message.

The routes selected by the hccE strategy for sending such aggregated messages are also illustrated in Figure 11. This shows how the strategy balances the packets' transmissions and the adaptivity of the routing protocol to find all possible paths between sources and sink.

D. Source-sink latency

Using the framework we can also determine the source to sink latency. In Figures 14-15 we illustrate comparatively the source to sink latency with aggregation enabled for the hc and hccE routing strategy, respectively. As expected, the source-sink latency is greater for the hccE strategy than for hc.

Since the first strategy selects also longer paths to balance the transmission load, the underlying T-MAC may need an extra active frame time (of 0,6s) until the data packets reach the sink.

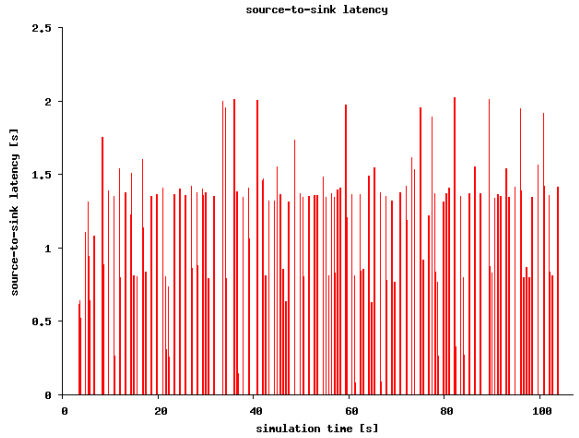

Fig. 14. hc: source to sink latency.

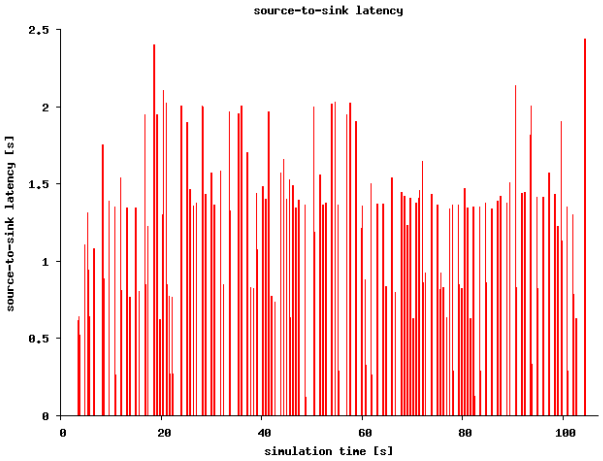

Fig. 15. hccE: Source to sink latency.

E. Latency MAC queue

Moreover, we can plot the latency for a packet in the MAC queue (namely the time between entering and leaving the queue) for each node, the number of packets in the MAC queue or (average) the number of collisions per node. For example, the latency for the aggregator (node 3) is illustrated in Figure 16.

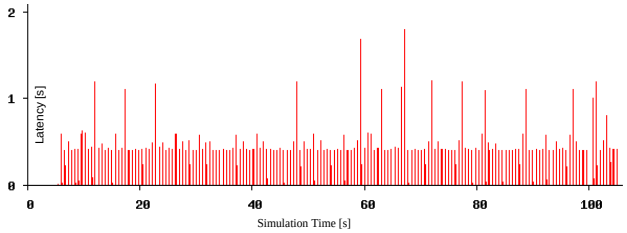

Fig. 16. Latency in MAC queue.

F. Impacts of data rate on the energy consumption

To illustrate the impact of data rate on the energy consumption in larger networks, we build another scenario with a test network consisting of 103 nodes with a distance of 9 hops (on the shortest path) between source and sink. At network layer we configure the simulator to route according to the hccE strategy. We first use T-MAC and set up three runs for different data intervals of 200ms, 500ms and 1000ms, respectively, which is equivalent to a data rate of 5 pkts/s, 2 pkts/s and 1 pkt/s. For T-MAC we set the listen time to 15ms and the frame time to 600ms. We repet the measurements for B-MAC, where the listen time is set to 50μs and the sleep time to 5ms.

Energy [mJ]	Data generation interval [ms]		
MAC Protocol	200	500	1000
T-MAC	273,48	238,08	213,67
B-MAC	264,33	220,65	197,75

TABLE IV
ENERGY CONSUMPTION USING DIFFERENT DATA GENERATION RATE.

Table IV gives the total energy consumption for both configurations with T-MAC and B-MAC.

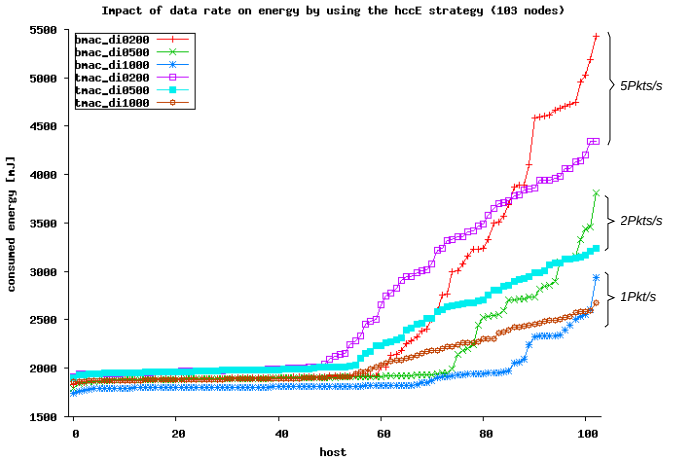

Fig. 17. Impact of using data rates on energy (hccE strategy).

Figure 17 illustrates the sorted energy consumption. If we compare the curves for a data generation interval of 200ms the energy consumption of the nodes on the path for B-MAC is still high, even though the total energy consumption is comparable with the one of T-MAC. By decreasing the data rate the consumption of the nodes on the path decreases considerably and B-MAC outperforms T-MAC.

G. Impact of unknown higher traffic depletion time

We consider the WSN as in Figure 11; additionally to sink 21 (data interval of 700ms, interest refresh is 5s) with 3 sources (right red rectangle) a new sink, node 41, was added which requests data from the zone containing node 2 (middle green rectangle) at a data interval of 500ms and sends refreshes at 4s. At MAC layer we use T-MAC protocol with listen time set to 60ms and frame time 600ms.
We set a very low initial energy for several nodes: $700mJ$ (equivalent to 3eU) for node 10 and $1000mJ$ (5eU) for nodes 0 and 7. The energy of nodes is converted in a scale between 0 and 255, which are called *energy units* (eU). Additionally we

multiply the energy consumption of the node (in all its states) with a factor of 10, to achieve a shorter simulation time. Such a situation could result from a long run of a previous interest, for example from sink 39, (or 41, 42) to sources 2, 7 and 10 (or subset of them).
The goal is to show the impact of the two strategies on the time when the nodes run out of energy (the depletion time).

By using the hc strategy, when both interests are active, the shortest route for the second interest goes through the nodes 7, 10, 0 which are depleted relatively fast by the first interest and the data packets are then obliged to travel along longer routes, e.g. 11-5-17-39-41. By using the hccE strategy the zone containg the nodes 0,7,10 is avoided. Since this is the shortest path from source 2 to sink 41 it is used for a short time at the beginning until the source gets information about its neighorhood (nodes 11 and 15). In our SNF it is easy to identify and visualize during simulation the route that a packet follows to reach the corresponding sink. Moreover, one can plot the number of data packets sent sent by the sources and the number of data packets received by the sinks in order to verify if all packets reach their destination. Under our scenario this is the case for both routing strategies.
The depletion times for the nodes configured with less energy are given in Table V.

Depletion time [s]	Nodes		
Strategy	Node 10	Node 0	Node 7
hc	40.32	61.27	74.43
hccE	55.87	76.81	82.85

TABLE V
IMPACT OF TRAFFIC AND STRATEGY ON DEPLETION TIME.

The hc strategy has preferred to route along shortest path and when the path was no longer available it used the next shortest path. On the other side, the hccE strategy has preferred to use longer paths in order to omit the nodes with lower energy reserve. Therefore the energy consumption by using the hc strategy must be smaller. It is noteworthy to remark that for this network scenario under the given settings the procentual difference between the total energy consumption of both strategies is below 2%, while the depletion time increases for the hccE strategy with a procentage between 12%-38% (achieved for the nodes 7 and 10, respectively).

The simulation and the results show that using the hccE strategy the lifetime of the nodes and thus of the network can be prolonged without significant penalties in the total energy consumption.

H. Impact of MAC

Besides comparing energy consumption of nodes, min/max latency in a node and between source and sink our framework allows also an analysis of the impact of a complete MAC protocol.

a) We investigate first the effects of different MAC protocols and their configuration parameters on the energy consumption of the nodes. Fig.18 illustrates the *sorted* energy consumption of all nodes using S-MAC and T-MAC protocols with two routing strategies, namely the hc and hccE. The energy is sorted in order to better visualize the distribution of the energy consumption on the nodes. The application requirements assume that the data interval generation is set to 200ms and the request is refreshed at a 5s interval. We use the following abbreviation, e.g., *Smac 60 600* employs S-MAC with an listen (active) time and frame time of 60ms and 600ms respectively and the hop count as default routing strategy, while *Tmac 15 600ce* uses T-MAC with an active time and frame time of 15ms and 600ms respectively and the hccE routing strategy.

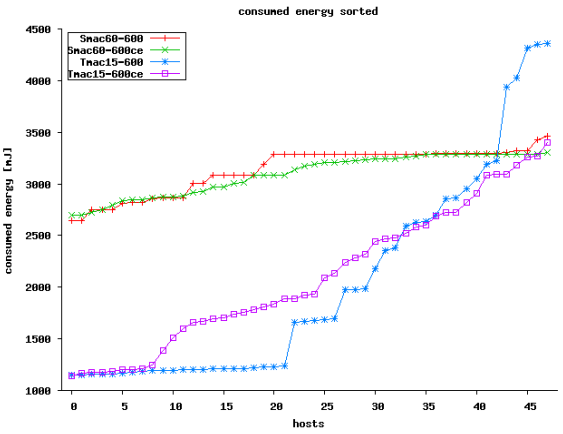

Fig. 18. Energy consumption using different protocol stacks (S-MAC and T-MAC with different parameter settings and two routing strategies).

It can be observed that in case of *Smac 60 600* (with the hc strategy) and *Smac 60 600ce* (with hccE), the influence of the strategy on the energy consumption is minimal, since the data traffic is relatively small and all the nodes are most of the time in idle listening and are consuming almost the same amount of energy. However, when employing *Tmac 15 600*, the choice between the hc and hccE strategy has a relevant impact on the energy consumption, the balancing policy of hccE is reflected by the smoother *Tmac15-600ce* curve than the *Tmac15-600* one for the hc strategy.

We chose here for S-MAC a higher active time than for T-MAC (60ms vs. 15ms) since for higher data rates S-MAC collapses (fails to deliver) than T-MAC as we will see next.

b) In the sequel we investigate the limits of the MAC protocols. The goal of this analyze is to find out when the MAC protocol is overloaded and is not able to deliver successfully data messages. We consider the MAC protocol overloaded if it discards more than 10% of the data messages. The broadcast messages (interest and its refreshes) are not taken into account, since we consider them not relevant for the application. The impact of different data rates on the behavior of the MAC (with a bounded queue) can be illustrated in Figure 19 by counting the total number of dropped frames.

Reasons to discard frames are either that the MAC queue is full or a transmission failure occurs (the maximal retries threshold to send the same frame was hit). One can observe that all MAC protocols have a point in the graphic from where the number of discarded frames increases steeply. If we consider the 10% limit as the point from where the MAC protocol is considered overloaded (unreliable), one can observe

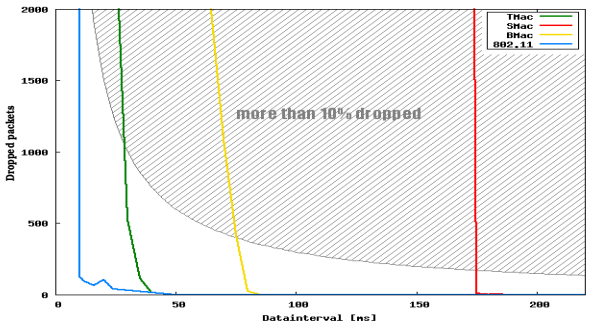

Fig. 19. Impact of the data generation interval on different MAC protocols.

that S-MAC hits this limit at a data generation interval of 180 ms (aprox. 5 frames/s), B-MAC at 75 ms (aprox. 13 frames/s) and T-MAC at 30 ms (aprox. 33 frames/s). Opposite to them, the WLAN 802.11 is characterized by a high efficiency, since even at 10 ms (aprox. 100 frames/s) it discards only few frames.

c) We study next what influence has the MAC protocol and the data generation interval on the source to sink latency. For time-critical sensor applications this behavior is an important aspect in choosing the appropriate MAC protocol. In order to analyze the latency we have taken 20 measurements with randomly placed sink and a constant distance of six hops between the source and sink. For different data generation intervals the results are illustrated in Figure 20.

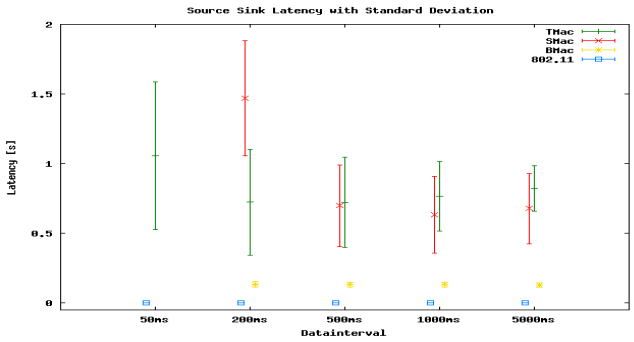

Fig. 20. Source-to-sink latency for different data rates and MACs.

In case of T-MAC the source to sink latency is relatively constant, increasing slightly at higher data rate (for 50ms), since the protocol nears to its collapse limit. The relatively high standard deviation shows that the number of active cycles that a packet needs to reach the sink is variable.
S-MAC experiences a relatively high latency (aprox. 1.5s) beginning with a data interval of 200ms, and collapses at a data interval of 50 ms (the protocol reaches its limits[5] see Figure 19). At lower data rate the latency is almost constant in the range of a frame time (0.6s). The number of hops that a packet travels per listen time is approximately fix and depends on the actual setting of this time. If the path has more hops than a packet can travel per listen time, it is cached in the MAC queue and waits the next listen period. That leads to a latency with a variation of one listen period.
B-MAC has a low latency without variation. This is due to its very low duty cycle compared to the one of S-MAC and T-MAC. Usually a packet needs here one (duty) cycle pro hop. The WLAN 802.11 has very low latency, without variation, since the protocol has no sleep state.

[5]Appropriate configuration of the listen period allows a correct function at 50 ms, but then the energy consumption increases

d) Finally, the dependency of the source-sink latency on the number of hops between the source and sink is illustrated in Figure 21. For this simulation the data generation interval is set to 500 ms and we take 20 measurements with a variable distance between source and sink in the range from 1 to 10.

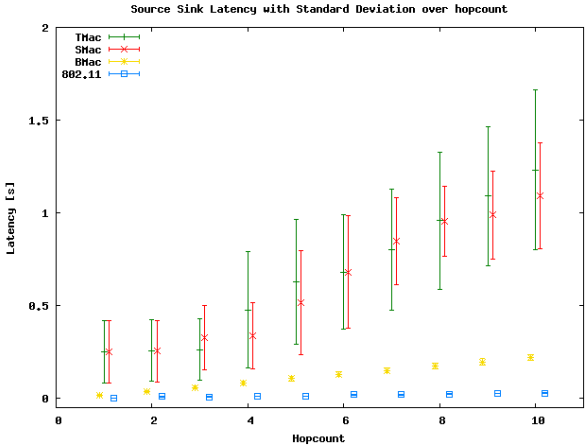

Fig. 21. Impact of the number of hops on the source-sink latency.

T-MAC and S-MAC have a similar behavior, they forward a packet 2-3 hops in one listen period and correspondingly the source-sink latency is small. After that the latency increases constantly with each new hop. Notice the slightly smaller latency (and variation) of S-MAC compared to T-MAC. The cause is the fact that at this lower data rate the listen time of T-MAC is seldom extended and therefore packets must wait a sleep period until the next listen period. In contrast, S-MAC is able to forward the packets more hops during its fixed listen period. B-MAC has a small, slightly increasing latency, while WLAN 802.11 has a very small latency even at higher hops.

As a concluding remark for the simulations involving the two phase multihop protocol we can state that both the choice of the MAC protocol and the choice of routing strategy with/without aggregation influence the energy consumption, and thus the network lifetime, significantly.

The same simulations and measurements can be carried over for the **enhanced directed diffusion (EDD)** protocol. Instead of presenting similar results here we choose to exemplify here the advantages of our decomposed, modular design of this complex routing protocol.

I. EDD scenario

To illustrate the exchange of a complete routing unit (RU) we implemented EDD in 5 RUs, each of them implementing one phase of the protocol, namely: interest propagation, path establishment, reinforcement, data event gathering and an additional RU to control the aggregation.

Here we want to illustrate the impact on energy consumption resulted by exchanging different RUs and SUs. We will exchanged, for example, for the first phase the flooding RU with a geographic flooding RU and for the third phase the strategy used at reinforcement. Instead of reinforcing the fastest neighbor that delivered the data event we chose the neighbor with the most residual energy able to deliver it.

For our simulation we use a more dense network (with more than 50 nodes, initially with a fixed residual energy of 2500 mJ), with node 0 as sink, placed in a central position of the network. The sink injects an interest requesting a data generation at 1s (refresh it at each 10s) in a zone with 8 sources, placed in the right side of the network as illustrated in Figure 22.

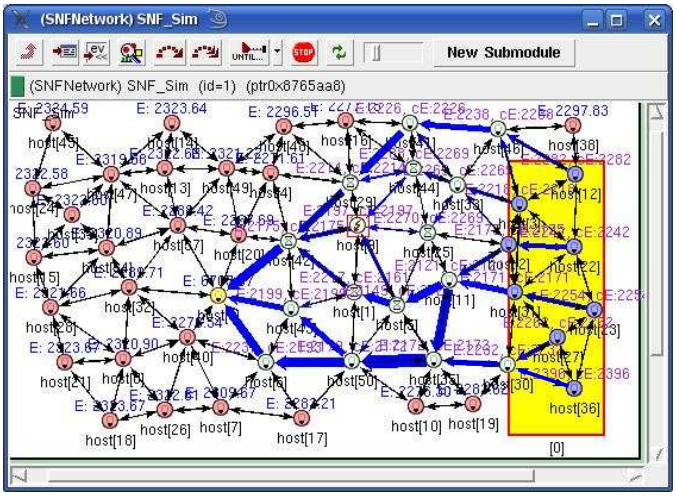

Fig. 22. Flooding and data flow routes at the end of simulation.

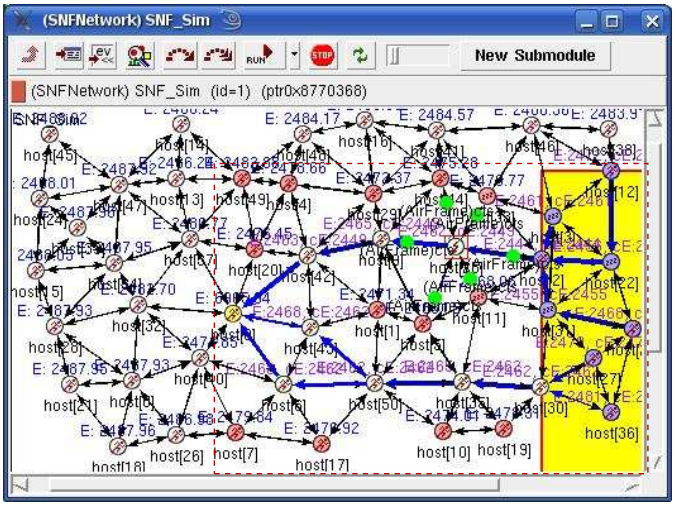

Fig. 23. Geographic flooding.

The interest is flooded in the whole network (each node that hears the interest is colored with light red). Figure 22 also shows the preferred paths (see the thick blue arrows[6]) that data packets have used to reach the sink. One can also remark that during the simulation several paths have been reinforced (the white nodes) due to energy consideration reasons.

We changed now the configuration by replacing the flooding used in the interest propagation phase with a geographic flooding unit (i.e., we just exchanged the corresponding RU). Since the network is dense enough the forwarding zone for the interest (and its refreshes) is restricted to the dotted rectangle determined by the sink and the destination zone (Figure 23).

As a result of our energy-aware strategy for the gradient reinforcement phase we obtain different data paths, namely the thick blue ones. Note that the data traffic is reduced, since the source 2 aggregates now the source nodes 3, 22 and 31, the last two aggregates source 12 and 33, respectively. Nodes 27 and 36 send as before their data directly.

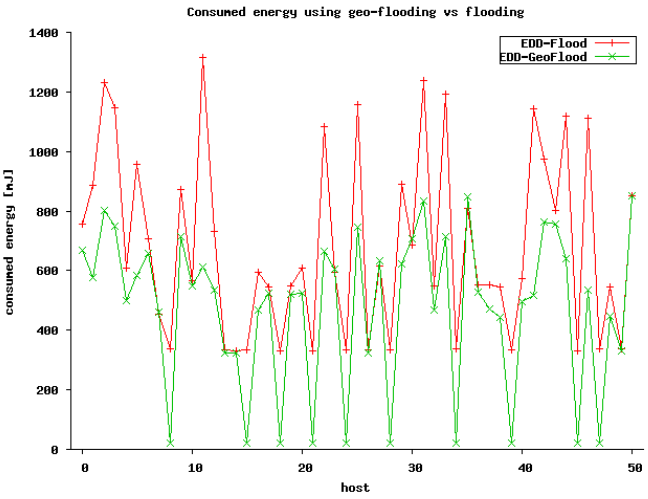

Fig. 24. Energy consumption by using flooding and geographic flooding.

The energy consumption for a 2 minutes run is plotted in Figure 24, where green stands for geographical flooding and red for flooding. Note the significant decrease in energy consumption in case of geographic flooding and also the fact that a lot of nodes have almost no energy consumption (the ones outside the forwarding region).

Of course, one can run simulations for enhanced directed diffusion in combination with different MAC protocols and illustrate all kinds of performance criteria (energy efficiency, latency, depletion time, collisions, etc) like in case of two phase routing protocol.

J. Simulator reconfiguration

The overhead required to reconfigure the simulator (in order to combine different building blocks of protocols) is small. For example, in order to modify a routing strategy we only need to

[6]the thickness of a link is according to the number of data packets that used that link

edit the strategy name. To activate the aggregation process one needs to edit the aggregation RU name and to set a flag. To exchange a complete MAC protocol one has to edit two lines in the configuration file of the node to include the new submodule and to import the corresponding code. Protocol's parameters (e.g., active and frame times) have corresponding parameters in the configuration file which can be edited. A recompilation step for the whole code is not necessary (assuming that all MAC protocols have been compiled) in order to start the simulation and to visualize the results.

VI. Conclusion and future work

In this paper we investigated several factors that impact on the performance of routing protocols used in resource constrained wireless sensor networks. The main performance criteria we are interested in are the energy consumption, the network lifetime and also the latency of the network in delivering replies to users requests. We analyzed the impact of factors such as various routing strategies, different MAC protocols and their configuration parameters, link and node failures, changes in the network topology, in-network processing, and fluctuating traffic. A node's behavior in terms of energy consumption is difficult to be exactly predicted, since it depends on a large amount of parameters, which may have adverse or unexpected effects, and thus also its impact on the evolution and performance of the entire network. Hence, an adequate modeling and simulation framework was needed in order to achieve a fine tuning of all these parameters and to better inspect the cumulative impact of their behavior on the sensor network.

To that aim we employed our SNF, a flexible tool to design and combine various protocols at application, network and MAC layers and to analyze the impact of different routing strategies and factors mainly on the energy consumption of the WSN. The framework can automatically visualize various performance criteria to enable a fast evaluation and comparison of protocols.

For routing protocols we proposed and compared several energy-aware routing metrics (§IV) by employing local and more global information concerning the residual energy of nodes. The main challenge here is to decide what is the relevant routing information that should be spread to the nodes in the network without sending explicit routing messages, in order to balance the load of forwarding the data packets on all the nodes by using different routing strategies. We exemplified this by describing two concrete protocols: a two phase multihop routing and enhanced, energy-aware directed diffusion.

We showed that an energy-aware routing can significantly contribute to better balance the communication load among nodes, and thus to prolong the network lifetime with minor penalties in the total energy consumption. Local neighborhood knowledge turned out to be insufficient to achieve this.

Aggregation is very useful to reduce the energy consumption of the nodes on the path and, additionally, it can reduce the traffic in the network avoiding in this way congestions and induced collisions (§V-C and §V-I).

Delaying the request (interest) broadcast is recommended to optimize further the energy consumption, but finding an optimal value is not trivial, especially when the underlying MAC protocol does not have a fixed schedule (§V-B). In such cases a closer exchange of information between the MAC protocol and the routing protocol is required. This can be accomplished by using the cross-layer component.

The simulation results have shown that the choice of the MAC protocol, especially its duty cycle, has a major impact on the energy consumption in the network. Thus, whenever the application requirements are known it is essential to select the MAC protocol appropriately. Moreover, the choice of the combination of the MAC and routing protocol influences the behavior of the network in terms of energy. An interesting observation is that the adaptive characteristic of the strategies combined with topology knowledge can be exploited when the MAC protocol is based on a preamble sampling scheme (like B-MAC) (see §V-F).

More important is the fact that the main impact on the energy consumption of the nodes is given by the MAC protocol and only secondary by the routing protocol.

Currently we provide implementations for different routing protocols and several low duty-cyle MAC protocols (S-MAC, T-MAC, Preamble Sampling), including support for collision detection and radio switch times).

As future work we intend to quantify the programming overhead needed to develop and integrate new protocols. Furthermore, since the MAC protocol has the main impact on the energy consumption, we intend to provide more MAC protocol implementations and to compare their performance. Carrying out comparative analysis between different MAC protocols and their interaction with network layer protocols will reveal surely other promising aspects that can bring optimization at both layers.

Additionaly, we strive for more modularity at MAC layer, mainly to embed at MAC layer more customizable services like the receiver-based contention (or other innovative ideas from new MAC protocols), which in our opinion gives another perspective to the interlayer communication and would improve the energy efficiency of routing.

References

[1] A. Kacsó and R. Wismüller, "Modeling and simulation of multihop routing protocols in wireless sensor networks," in *Proc. 5th Int. Conf. on Wireless and Mobile Communications (ICWMC'09)*. Cannes, France, August 2009, pp. 296–302.

[2] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion a scalable and robust communication paradigm for sensor networks," in *Proc. ACM MobiCom*. Boston, 2000, pp. 56–67.

[3] F. Ye, A. Chen, S. Lu, and L. Zhang, "A scalable solution to minimum cost forwarding in large sensor networks," in *Proc. 10th Int. Conf. on Comp. Comm. and Networks*. Arizona, 2001, pp. 304–309.

[4] F. Ye, G. Zhong, S. Lu, and L. Zhang, "Gradient broadcast: A robust data delivery protocol for large scale sensor networks," *Wireless Networks/Springer, The Netherlands*, vol. 11, no. 2, pp. 285–298, 2005.

[5] M. Busse, T. Hänselmann, and W. Effelsberg, "Energy-efficient forwarding schemes for wireless sensor networks," in *Proc. Int. Symp. on WoWMoM*. New York, USA, June 2006, pp. 125–133.

[6] C. Schurgers and M. Srivastava, "Energy efficient routing in wireless sensor networks," in *Proc. MILCOM on Comm. for Network-Centric Operations: Creating the Inform. Force*. Virginia, 2001, pp. 357–361.

[7] H. Nurul1, M. Hossain, S. Yamada, E. Kamioka, and O.-S. Chae, "Cost-effective lifetime prediction based routing protocol for manet," in *Proc. of ICOIN*. Springer-Verlag Berlin, 2005, pp. 170–177.

[8] M. Maleki, K. Dantu, and M. Pedram, "Lifetime prediction routing in mobile ad-hoc netowrks," *Proc. IEEE WCNC*, vol. 2, pp. 1185–1190, March 2003.

[9] J. Aslam, Q. Li, and D. Rus, "Three power-aware routing algorithms for sensor networks," *Wireless Comm. and Mob. Computing*, vol. 3, no. 2, pp. 187–208, 2003.

[10] E. Shih, S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *Proc. 7th Ann. Int. Conf. on Mob. Comp. and Netw.* Rome, Italy, July 2001, pp. 272–287.

[11] Y. Yu, R. Govindan, and D. Estrin, "Geographical and energy aware routing: a recursive data dissemination protocol for wsns," in *UCLA/CSD-TR-01-0023s*. LA, California, USA, May 2001, pp. 171–180.

[12] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2005.

[13] G. Halkes, T. Dam, and K. Langendoen, "Comparing energy-saving mac protocols for wireless sensor networks," *Mob. Netw. Appl.*, vol. 10, no. 5, pp. 783–791, 2005.

[14] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *Proc. IEEE SECON*. Santa Clara, CA, October 2004.

[15] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," in *Proc. 2-nd ACM Int. Symp. on Mobile ad hoc networking and computing*. Long Beach, CA, USA, October 2001.

[16] V. Bharghavan, A. Demers, S.Shenker, and L. Zhang, "Macaw: A media access protocol for wireless lans," in *Proc. of SIGCOMM Conf.* London, UK, September 1994, pp. 212–225.

[17] A. Kacsó and R. Wismüller, "A framework architecture to simulate energy-aware routing protocols in wireless sensor networks," in *Proc. IASTED Int. Conf. on Sensor Networks*. Greece, 2008, pp. 77–82.

[18] A. Varga, *User Manual*. OMNeT++ Version 3.2, 2006.

[19] M. Löbbers and D. Willkomm, *Mobility Framework for OMNeT++ (API ref.)*. http://mobility-fw.sourceforge.net: OMNeT++ Ver.3.2, 2006.

[20] A. Kacsó and R. Wismüller, "A simulation framework for energy-aware wireless sensor network protocols," in *Proc. 18th Int. Conf. on Computer Communications and Networks, Workshop on Sensor Networks*. San Francisco, CA, USA, August 2009.

[21] J. Polastre, J. Hui, P. Levis, J. Zhao, D. Culler, S. Shenker, and I. Stoica, "A unifying link abstraction for wireless sensor networks," in *Proc. 3rd ACM Int. Conf. SenSys*, November 2005, pp. 76–89.

[22] C. Ee, R. Fonseca, S. Kim, D. Moon, A. Tavakoli, D. Culler, S. Shenker, and I. Stoica, "A modular network layer for sensornets," in *Proc. 7th Symp. OSDI*. Seattle, WA, USA, 2006, pp. 249–262.

[23] I. Akyildiz, M. Vuran, and O.Aka, "A cross-layer protocol for wireless sensor networks," in *Proc. CISS*. Princeton, NJ, March 2006.

[24] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated, adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. on Netw.*, vol. 12, no. 3, pp. 493–506, 2004.

[25] T. Dam and K.Langendoen, "An adaptive energy- efficient mac protocol for wireless sensor networks," in *Proc. 1st Int. Conf. on Embedded Networked SenSys*. LA, California, USA, 2003, pp. 171–180.

[26] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. 2nd ACM Int. Conf. on Embedded Networked SenSys*. NY, USA, November 2004, pp. 95–107.