Efficient Energy-Aware Mechanisms for Real-Time Routing in Wireless Sensor Networks

Mohamed Aissani, Sofiane Bouznad, Badis Djamaa, and Ibrahim Tsabet

Computer Science Unit, Ecole Militaire Polytechnique (EMP)
P.O. Box 17, Bordj-El-Bahri, 16111, Algiers, Algeria
{maissani,bouznad.sofiane,badis.djamaa,utopia.ibrahim}@gmail.com

Abstract. We propose three mechanisms to manage nodes energy and improve the efficiency of real-time routing protocols in sensor networks. To preserve nodes' resources and to improve network fluidity, the first mechanism removes each useless packet due to its insufficient deadline in reaching the sink. To reinforce the packet real-time aspect, the second mechanism selects from the current-node queue the most urgent packet to be forwarded first. For a better node energy balancing, the third mechanism uses both the residual energy and the relay speed of the forwarding candidate neighbour to select the next forwarder of the current packet. These mechanisms are simple to implement, require very little states and rely only on local primitives. In addition they can be easily integrated in any geographic routing protocol. Associated with the real-time routing protocol SPEED in TinyOS and evaluated in the simulator TOSSIM, our proposals achieved good performance in terms of node energy balancing, packet loss ratio and energy consumption.

Keywords: Wireless sensor networks, real-time routing, energy-aware routing, node energy balancing.

1 Introduction

Wireless sensor networks (WSNs) are often characterized by a dense and large scale deployment of battery-powered sensors with limited processing, storage and communication resources. It is widely recognized that conserving energy is a key requirement in the design of WSNs, because of the strict constraints that imposes on network operations [1, 2, 3]. In fact, the energy consumed by sensor nodes has an important impact on the network lifetime that has become the dominant performance criterion. Extending lifetime of a WSN is a shared objective by WSN designers and researchers across the whole network stack. At the routing layer, for instance, it is necessary that routing algorithms use less energy-consuming paths; this challenge may be aggravated in real-time routing protocols where the real-time quality of service (QoS) imposes more constraints on these paths.

To address this issue, we propose in this paper, three energy-aware mechanisms for real-time geographic routing protocols in WSNs. While strictly respecting the real-time constraint of the routed packets, the proposed mechanisms address both network energy consumption and node energy balancing. The first mechanism, called UPR (Useless Packet Removing), removes any packet considered unnecessary since it has no chance to reach its destination before the expiration of its deadline. The second mechanism, called UPS (Urgent Packet Selecting), selects, from the current-node queue, the most urgent packet to be forwarded first. The third mechanism, called CAB (Cost-Aware energy Balancing), uses both the residual energy and the relay speed of a forwarding candidate neighbor to select the next forwarder of the most urgent selected packet.

The rest of the paper is organized as follows. Section 2 summarizes the related work. Section 3 describes the proposed energy-aware mechanisms (UPR, UPS and CAB) that improve the performance of real-time geographic routing protocols in WSNs. Section 4 presents evaluations and discussions of the performance of our proposals. Section 5 concludes the paper and discusses future research directions.

2 Related Work

Some existing real-time routing protocols [4, 5, 6, 7] define explicit mechanisms to save energy of sensor nodes and/or maximize the network lifetime. PATH [4] improves real-time routing performance by means of reducing packets' dropping in routing decisions. It is based on the concept of using two-hop neighbor information and power-control mechanism. The former is used for routing decisions while the latter is deployed to improve link quality as well as reducing the delay. The protocol dynamically adjusts transmitting power in order to reduce the probability of packet dropping and addresses practical issue like network holes, scalability and loss links in WSNs. EARTOR [5] is designed to route requests with specified end-to-end latency constraints, which strikes the elegant balance between the energy consumption and the end-to-end latency and aims to maximize the number of the requests realized in the network. The core techniques adopted include the cross-layer design that incorporates the duty cycle, a bidding mechanism for each relay candidate that takes its residual energy, location information, and relay priority into consideration. EEOR [6] improves the sensor network throughput by allowing nodes that overhear the transmission and are closer to the sink to participate in forwarding the packet, i.e., in forwarder list. The nodes in the forwarder list are prioritized and the lower priority forwarder will discard the packet if the packet has been forwarded by a higher priority forwarder. One challenging issue is to select and prioritize forwarder list such that the energy consumption by all nodes is optimized. Extensive simulations show that this protocol performs well in terms of energy consumption, packet loss ratio and average delivery delay. TREE [7] is a routing strategy with guaranty of QoS for industrial wireless sensor networks by considering the real-time routing performance, transmission reliability, and energy efficiency. By using the two-hop information, the real-time data routes with lower energy cost and better transmission reliability are used in the proposed routing strategy.

Although the previous works contribute to improving network performance, design of energy-aware real-time routing protocols in WSNs is still a challenging issue. Consequently, three efficient energy-aware mechanisms (UPR, UPS and CAB) are proposed in this paper and then associated with the well-known real-time routing protocol SPEED [8], which gives birth to the CA-SPEED (Cost Aware SPEED) protocol. Note that SPEED [8] is designed to be a stateless, location-based routing protocol with minimal control overhead. It achieves an end-to-end soft real-time communication by maintaining a desired delivery speed across the sensor network through a novel combination of feedback control and stateless non-deterministic geographic forwarding.

3 Proposed Energy-Aware Mechanisms

We consider a WSN comprising a set of homogenous nodes N, arbitrary located in the plane. A node in the system is a device which has a processing unit, a power unit and a communication unit allowing wireless communication. For the communication model, we take the protocol model, in which all nodes located within the transmission range r of a given node i, form the set of i's potential neighbors, thus the links are assumed to be bi-directional. The combination of the nodes N and the neighborhood relations E form a WSN, which can be represented by an undirected graph $G = (N, E)$. In our simulations (Section 4), a more realistic physical model which includes a realistic signal propagation model, signal interference, distortions, background noise, etc. was used.

Since our problem consists of proposing efficient energy managing and latency improving mechanisms for real-time geographic routing protocols in WSNs, two other principal assumptions made in this category are considered. First, it is assumed that every node knows its own and its network neighbors' positions. Second, the source of a message is assumed to be informed about the position of the destination. The Knowledge of nodes' and sinks' positions is assumed to be reasonable [9]. By regards, to the neighbors' positions, this is realized by a neighbor discovery mechanism (known as location beacons) implemented in the majority of geographical routing protocols including SPEED, in which nodes periodically broadcast location beacons packets containing their positions to their neighbors. Neighbors receiving such a packet store the contained information in their caches for a specific time period. Given a data packet p to be transmitted from a node u to a node v, in one- or multi-hop way, we use the following notations: $D_{uv}(p)$ is the Euclidean distance between the nodes u and v; $T_{uv}(p)$ is the delay that the packet realized from node u to node v or the expected delay the packet should register to get from u to v; $Deadline(p)$ is the packet deadline; $AD(p)$ is the packet advance in distance from its source u w.r.t the destination node v; $AT(p)$ is the packet advance in time at current-node u. It will be measured based on the packet deadline and the expected delay ($T_{uv}(p)$) that the packet should register to get from current-node u to destination v; $H_{uv}(p)$ is the number of hops the packet travelled or ought to travel from node u to node v.

3.1 UPR Mechanism

Many existing real-time routing protocols, such those summarized in Section 2, forward, often over long distances, data packets that may have no chance to reach their destination node because of their insufficient residual deadlines resulting in network resource wasting (throughput and energy). This is because those protocols do not exploit the packet deadline information to decide on the utility of a packet. Starting from this point and in order to save nodes precious resources, the proposed UPR mechanism ensures an early removal of any useless packet that will not reach its destination. Indeed, only packets with sufficient residual deadline to reach their destinations are forwarded in the network. Thus, UPR saves nodes energy and increases the network fluidity which reinforces the real-time aspects of the associated routing protocol. To do so, UPR estimates the expected end-to-end delay allowing the current packet to reach its destination node (Section 3.1.1), then applies a decision rule, based on both the expected end-to-end delay and the application time/energy requirements, expressed by an α parameter, to decide whether to remove or forward the packet (Section 3.1.2).

3.1.1 Expected End-to-End Delay

Since the end-to-end delay in a multi-hop networks depends on the distance that a packet travels, many real-time geographic routing protocols such as SPEED, and its derived protocols, route packets according to the packet's maximum delivery speed, defined as the rate at which the packet should travel along a straight line to the destination [10].

Exploiting this feature, we propose a mechanism allowing a node in the network to estimate the expected end-to-end delay for a current packet, based on the already known past registered delay tailored to the already travelled end-to-end distance and an expected end-to-end distance which the packet should travel along a straight line from the current node to the destination. For coherency reasons, the already travelled end-to-end distance is taken as the straight distance realized by the packet.

Formally, we propose in this section a mechanism, depicted in Formula (1), which allows a current-node i to estimate the expected end-to-end delay for a packet p to reach its destination d, which is denoted by $T_{id}(p)$ in Fig. 1(a). The proposed mechanism estimates $T_{id}(p)$ as a function of the already known delay, registered by p from its source node s (dubbed $T_{si}(p)$ in Fig. 1(a) and Formula (1)), tailored to a ratio between the end-to-end distance that the packet p travelled from its source node s to current-node i (dubbed $D_{si}(p)$) and the remaining end-to-end distance for the packet to reach its destination (dubbed $D_{id}(p)$).

$$T_{id}(p) = \frac{D_{id}(p)}{D_{si}(p)} * T_{si}(p) \tag{1}$$

In addition to its simplicity, the estimator proposed in Formula (1) bases its calculus only on local information either in current packet p or in current-node i. To calculate the Euclidian distances, the current node needs only to know, in addition to its position, the positions of the source and destination nodes which are incorporated in data packets transmitted by geographic routing protocols (the source position is used to indicate the area of the reported event and the destination position is used in routing decisions). The already registered delay is derived from the deadline field, which is an intrinsic propriety of real-time data packets. Being based only on local information, our estimator is robust to topology changes; network density and more importantly scales well with network size. Note that since the proposed estimator reposes on past registered delay by the packet, it takes into account all the parameters caused this delay (processing, queuing, etc.) and inject them to estimate the expected future delay.

In addition to the above characteristics, the proposed estimator in Formula (1) seems to give best-case expected end-to-end delays at the beginning of the packet transmission, which allows it to get in the network acquiring more knowledge and hence getting more accurate end-to-end delay expectations. Thus the expected end-to-end delays approach reality, when the message travels further in the network, making deletions based on it more accurate. However, in some extreme cases, the expected end-to-end delay given by our estimator may: (1) Exceed the real delay (if we allow forwarding the packet) which may compromise decisions based on our estimator or; (2) Underestimate the real delay allowing a useless packet to be forwarded when it should be deleted, which causes resources wasting.

To illustrate the former scenario, take a case where in the first travelled part, the packet registered relatively large delays which make the estimator expect a large future delay for the remaining distance to travel. However, if the remaining network portion is uncongested, the packet will register less delay (over estimation); therefore there is a big risk to delete a real-time data packet which is more probably to reach its destination. As an example of the latter, take the opposite extreme case when the first network portion is relaxed and the second one is congested, then the remaining delay may be underestimated and a packet which should be deleted earlier will do more steps, which induce more energy consumptions. This analysis is taken into account in designing the UPR packet-removing algorithm.

3.1.2 Packet-Remove Decision

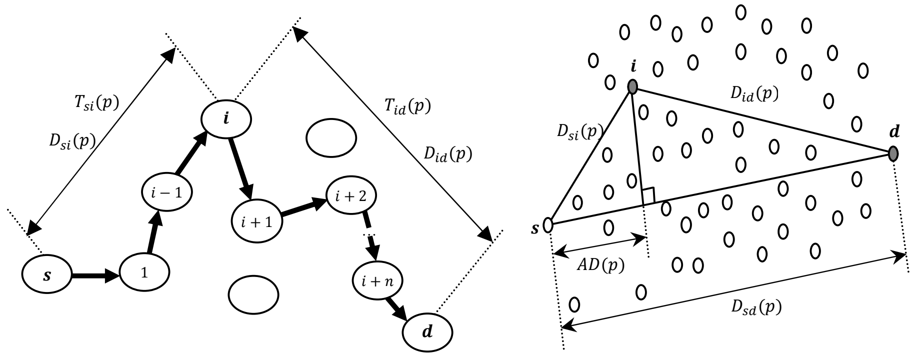
Erroneous deletion of real-time data packets based on the above estimator could be devastating for time-critical applications. To avoid this problem, we propose a packet removing algorithm that tailors the expected delay given by the estimator to the application time/energy requirements specified by the α parameter. Hence, our removal algorithm automatically and dynamically calibrates the estimated delays based on the packet advance in distance towards its destination, until a specific ratio defined by the α parameter after which the energy-requirement is given more weight.

As explained above, the proposed estimator gives more realistic, hence trusted, values when the packet being acquired more past knowledge in advancing towards its destination. This advance could be presented by the distance gain realized by the packet from its originator node s to the current-node i w.r.t the total distance between the source s and the destination d. For this reason, and to give more chance to the packet, in time critical-applications, the proposed packet-removal algorithm dynamically adjusts the expected end-to-end delay by a dynamic factor, in the interval [0, 1], based on the advance in distance realized by the packet, until a given ratio imposed by the application energy-requirements, after which a delayed packet is automatically removed. This ratio is expressed and fixed by the αparameter defined by the application, depending on its energy/time requirements. Thus, theαparameter decides on the proportion in distance advancement after which the application will prefer the energy-saving requirement against the time-saving one. Before the α ratio, the time-requirement is more weighted, thus estimated delays are calibrated to give the packet more chances.

Formally, having the expected end-to-end delay $T_{id}(p)$ (see Formula (1)) and to decide whether to remove or forward packet p, current-node i applies Decision-rule (2), where $D_{sd}(p)$ denotes the distance between source node$s$ and destination node$d$, $AD(p)$ shown in Fig. 1(b) and given by Formula (3) represents the advance in distance of packet p toward its destination node, and α is the energy/time parameter set in the interval [0,1] according to the real-time application requirements. The thresholdα must be close to 0 in energy-critical applications maximizing the network lifetime and close to 1 in time-critical applications minimizing the packet loss ratio of the used routing protocol.

Decision-rule (2) is explained as follows. If $AD(p)$ is greater than $\alpha * D_{sd}(p)$ then node i prefers saving nodes energy and hence removes each delayed packet. Otherwise (i.e. before reaching the threshold α), $T_{id}(p)$ is calibrated and multiplied by a dynamic factor less than 1: $AD\,(p)/\big(\alpha * D_{sd}(p)\big)$ to give more chance to the packet p to advance toward its destination d. If the result exceeds $Deadline(p)$ despite the given chance then packet p is removed to save network resources and to increase the network fluidity. In our simulations (Section 4), parameter α is set to 0.5. Thus, each delayed packet p with an advance $AD(p)$ greater than 50% of the total distance $D_{sd}(p)$ is immediately removed by current-node i in order to increase fluidity of links and to save the limited energy of sensor nodes.

```
IF $AD(p) > \alpha * D_{sd}(p)$ THEN
        IF $T_{id}(p) > Deadline(p)$ THEN remove packet $p$;
        ELSE packet $p$ is to forward;
        ENDIF
    ELSE                                                        (2)
        IF$\frac{AD(p)}{\alpha * D_{sd}(p)} * T_{id}(p) > Deadline(p)$ THEN remove packet $p$;
        ELSE packet $p$ is to forward;
        ENDIF
```



(a) Current-node i calculates $T_{id}(p)$ (b) Current-node i calculates $AD(p)$

Fig. 1. The UPR mechanism: node i decides on the removal (or the forwarding) of current packet p

To express the advancement in distance $AD(p)$ for a packet p, we apply the Pythagorean Theorem to two rectangular triangles depicted in Fig. 1(b). resulting in Equation (3) below.

$$AD(p) = \begin{cases} \frac{D_{si}^2(p) - D_{id}^2(p) + D_{sd}^2(p)}{2D_{sd}(p)} & \text{IF} \quad D_{si}(p) < D_{sd}(p) \\ D_{sd}(p) & \text{OTHERWISE} \end{cases} \tag{3}$$

3.2 UPS Mechanism

Most existing real-time routing protocols use scheduling schemes based only on the residual deadline of a packet to be forwarded [11]. However, these schemes may not be effective when packets are sent to different destinations; case of a network using several sinks. In fact, these schemes prioritize forwarding a packet whose deadline is the smallest although it is very close to its destination. To illustrate this point, Fig. 2 depicts an example in which current-node i has two packets to forward: p_1 for destination $d1$ with 2 ms (milliseconds) as deadline and p_2 for destination $d2$ with 3 ms as deadline. According to existing scheduling schemes based only on the residual deadline, node i will firstly forward p_1 and then probably will removep_2 because of its distance to destination $d2$. To provide an efficient solution to this problem, the proposed UPS mechanism combines both the residual deadline and the expected end-to-end delay of a packet in a decision parameter $D(p)$to decide on its urgency. In other words, the UPS decision parameter schedules packets based on their gained-deadline to reach their destinations, expressed as the difference between the packet residual deadline and its expected end-to-end delay to reach its destination. Formally, UPS performs as follows: (a) For a data packet p_j in the queue of current-node i, calculate the decision parameter $D(p_j)$ by Formula (4), where $T_{id}(p_j)$ is the expected end-to-end delay, calculated by Formula (2), allowing packet p_j to reach its destination d; and (b) Selects data packet p_k having the smallest decision parameter $D(p_k)$ by running Function (5).

$$D(p_j) = Deadline\ (p_j) - T_{id}(p_j) \tag{4}$$

$$D(p_k) = Min\{D(p_j)\ ;\ \forall p_j \in Queue(i)\} \tag{5}$$

Using the UPS mechanism, the two packets p_1 and p_2 (Fig. 2) will probably reach their respective destination before their deadlines expire as current-node i will forward packet p_2 before packet p_1. Note that UPS is also valid for a network using one sink (destination node).

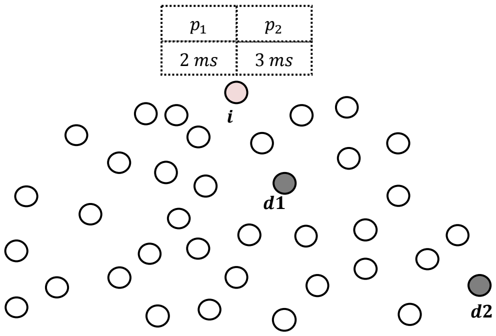


Fig. 2. Selection of the most urgent packet from the queue of current-node i

The urgent packet selection mechanisms can be performed in two ways: (a) during the packet reception by a node where its queue is scheduled according to Formula (4) or (b) during the forwarding process where the most urgent packet is timely chosen from the current-node queue. By using way (a), UPS minimizes calculations but loses reliability because the queuing time of packet p_j is not considered when estimating its expected end-to-end delay $T_{id}(p_j)$. But by adopting way (b), the current-node queue is not scheduled and selection of the most urgent packet requires extraction of all packets belonging to this queue. Since UPS is designed to achieve lower loss ratio and energy efficiency, we implemented way (b) where a current node applies both Decision-rule (2) and Function (5) on all packets in its queue, in order to remove each delayed packet and to select the most urgent packet among the not delayed ones.

3.3 CAB Mechanism

To choose the best next forwarder in terms of residual energy and relay speed, the current node uses the CAB mechanism, which is based on two efficient algorithms:

neighbor energy estimation (Section 3.3.1) and next forwarder selection (Section 3.3.2). This way, CAB aims to deliver a maximum number of real-time packets with a good node energy balancing in order to maximize the network lifetime.

3.3.1 Neighbor Energy Estimation

In order to allow nodes to be aware of their neighbors' residual energies, we propose a mechanism allowing nodes to exchange this information. In fact, each node keeps in its neighbor table a $ResidualEnergy$ field representing the residual energy of each neighbor. The update of this field is discussed below. Location beacons, used in many geographical routing protocols, seem to be a good way to exchange residual energy information between nodes of a WSN. However, since location beacons are not sent frequently in static topologies, they are not sufficient to update energy information. This drives us to include the $ResidualEnergy$, in each forwarded packet. Relying on this field, the receiving node updates its own neighbors table. In this solution, the most energy updates are carried out through the location beacons that are more regular and sent in a broadcast mode to all neighbors of a node. Data packets, which are generally routed over long distances, consolidate these updates. Acknowledgments packets, sent after receiving data packets, improve the efficiency of our energy estimator.

This solution is very reliable in both static and dynamic environments. Thus, in mobile networks location beacons are sent frequently and hence updates are regular which increases routing decisions efficiency in our CAB mechanism. For lightly loaded static networks, location beacons are less frequent, which leads to less energy updates. However, in less active networks, energy consumption is reduced, so that frequency of energy updates is proportional to both network activity and energy consumption.

3.3.2 Next Forwarder Selection

We use the same definition of FS (Forwarding candidate neighbors Set) proposed in SPEED (Fig. 3), but we propose improved strategies to select next forwarder of packet p. In SNGF (Stateless Nondeterministic Geographic Forwarding) component of SPEED [8], the next forwarder of current packet p toward the destination node is selected by current-node i from its FS, which is constructed from its NS (Neighbors Set), according to a relay speed metric. Node n_i is a forwarding candidate if $distance(n_i, d) < distance(i, d)$. In Fig. 3, $FS = \{n_1, n_2, n_3\}$ and $NS = \{n_1, n_2, n_3, n_4, n_5, n_6, n_7\}$. The relay speed provided by a neighbor n_i in FS is given in SPEED by Formula (6), where L is the distance between current-node i and destination d, $Lnext$ is the distance between neighbor n_i in FS and destination d, and $HopDelay(n_i)$ is the estimated delay of the link $i \rightarrow n_i$.

$$S(n_i) = \frac{L - Lnext}{HopDelay(n_i)} \quad ; \quad n_i \in FS \tag{6}$$

Our goal is to insure a good node energy balancing without affecting the real-time aspect of the routed flows in a WSN; i.e. to achieve a lower packet loss ratio and a higher energy balancing factor. To select the next forwarder, the proposed CAB mechanism combines both the relay speed and the residual energy of all neighbors in FS. Neighbor $n_k$with the largest decision parameter $D(n_k)$is selected by current-node i as next forwarder. Formally, node i applies Formula (7) that uses $D(n_k)$ given by Formula (8), where $S(n_i)$ is the relay speed provided by neighbor n_i, $E(n_i)$ is the residual energy of n_i, S_{max} is the maximal relay speed in FS, E_{max} is the maximal residual energy in FS, and f is the factor speed/energy varying in the interval [0,1].

The weighting factor f is dynamically adjusted in a way that considers the network energy-balancing only when the message has realized an advance in its time-requirement allowing the node to choose the neighbor that better balances the energy consumption without disrupting the application real-time exigencies. Its adjustment is tightly related to the α parameter, representing the application energy/time requirement. Formally, the weighting factor f is obtained using Formula (9), where $H_{id}(p)$ denotes the remaining hops of packet p that is given by Formula (10), $H_{si}(p)$ is the number of hops traveled by p until node i (read from the packet TTL field), $AT(p)$ is the packet advance in time given by Formula (11), $AD(p)$ and $T_{id}(p)$ are given by Formulas (3) and (1) respectively.

$$NextHop(p) = n_k \,; \quad \text{with}: D(n_k) = Max\{D(n_i)\,;\ \forall n_i \in FS\} \tag{7}$$

$$D(n_i) = f * \frac{S(n_i) - S_{min}}{S_{max} - S_{min}} + (1-f) * \frac{E(n_i) - E_{min}}{E_{max} - E_{min}} \quad ; \quad n_i \in FS \tag{8}$$

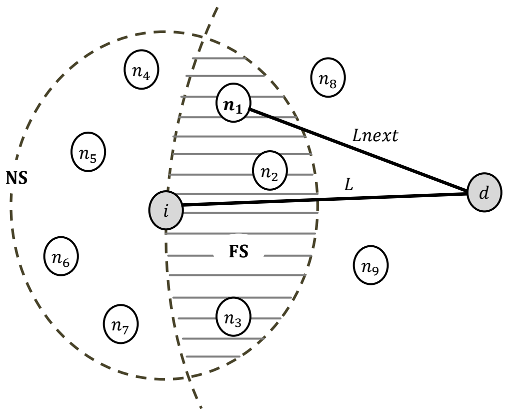


Fig. 3. Sets NS and FS of current-node i in the SPEED real-time routing protocol

Formula (9) is explained as follows: before reaching the threshold α where the expected end-to-end delay may be underestimated, the weighting factor f prioritizes the relay speed allowing the message to get forwarded and increasingly weights the residual energy in the forwarding decision, when the packet realizes an advance in distance, w.r.t to the destination, which is expressed by $AD(p)/D_{sd}(p)$ in Formula (9). After the threshold, the CAB mechanism always prioritizes the rely-speed unless the packet has gained time and has fewer hops to go. This is expressed by the $AT(p)/H_{id}(p)$ ratio in Formula (9).

$$\begin{cases} 1 - \dfrac{AT(p)}{H_{id}(p)} & \text{IF } AD(p) > \alpha * D_{sd}(p) \\ \\ 1 - \dfrac{AD(p)}{D_{sd}(p)} & \text{OTHERWISE} \end{cases} \tag{9}$$

$$H_{id}(p) = D_{id}(p) * \frac{H_{si}(p)}{D_{si}(p)} \quad (10)$$

$$AT(p) = \frac{Deadline(p) - T_{id}(p)}{Deadline(p)} \quad (11)$$

The above behavior of the weighting factor $f$makes it always preferring the rely speed on the energy balancing. It only exploits the gain in time realized by the packet and the application energy/time requirements to explicitly enhance the protocol energy balancing; thus allowing the CAB mechanism to enhance network lifetime, without disturbing the application real time real-time exigencies.

4 Performance Evaluation

To evaluate the performance of the proposed mechanisms (UPR, UPS and CAB), we associated them with the well-known real-time routing protocol SPEED [8] and the resulting protocol is called CA-SPEED (Cost-Aware SPEED). We changed only the SNGF component of SPEED. In CA-SPEED, when a current node has to forward a data packet it: (1) Removes all delayed packets from its queue by executing the UPR mechanism and selects, during this queue consultation, the most urgent packet to forward first by using the UPS mechanism; and (2) Forwards the selected urgent packet to its best neighbor node which is obtained by executing the CAB mechanism.

The protocols SPEED and CA-SPEED were implemented in TinyOS [12] and evaluated in its embedded sensor network simulator TOSSIM [13]. Also, the recent proposed routing protocol EEOR [6] was evaluated in this simulator in the same conditions. Since we are interested in real-time applications, we used a scenario of detecting events that occur randomly in a field of interest. Once an event is detected, the information captured will be forwarded in a required deadline towards a sink, which is usually connected to an actuator. Our simulation scene uses a uniform random distribution of sensor nodes. We performed simulations on a 500×500 meters terrain with 625 deployed sensor nodes (an average density of 12 neighbors per node). Two destination nodes are deployed and each one receives packets concerning particular event detection. At each time period, 20 randomly source nodes, equitably distributed on each side of the network, detect an event and forward corresponding information to one destination node (sink). Each simulation runs for 230 seconds. Table 1 summarizes the parameters used in our simulations.

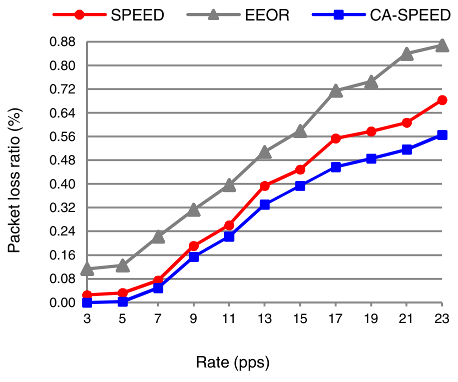
Table 1. Simulation parameters

Parameter	Value
MAC layer	CSMA-TinyOS
Radio layer	CC2420 radio layer
Propagation model	log-normal path loss model
Queue size	50 packets
Transmission channel	WirelessChannel
Bandwidth	200 Kilobytes per second
Packet size	32 bytes
Energy model	PowerTOSSIMz model
Node radio range	40 meters

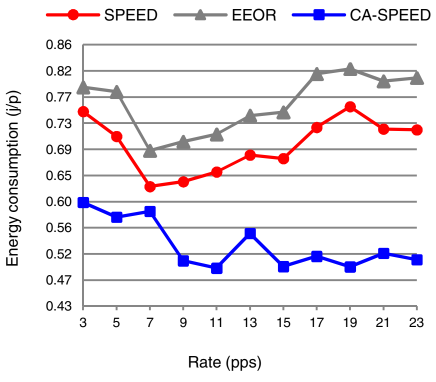
For each run, we measure the packet loss ratio, energy consumption per delivered packet and energy balancing factor (ebf). The later represents the variance in energy consumed by all sensors having the same initial-energy quantity. Formally, we have $ebf = (1/ns) * \sum_{k=1}^{ns}(ec_k - ec_{avr})^2$, where ec_k is the energy consumed by sensor k and ns is the number of deployed sensors, ec_{avr} is the average energy consumed by all deployed sensors.

To measure the impact of packet generation rate, resulting from event detection applied in the source nodes, when generating real-time packets on each protocol performance, we set the packet deadline to 500 ms and we vary the source rate from 3 to 23 pps (packets per second). For each simulation, we measure performance achieved by the protocols SPEED [8], EEOR [6] and CA-SPEED. Obtained simulation results, shown in Fig. 4, illustrate that our energy-aware mechanisms, used in the CA-SPEED protocol, are efficient in terms of delivering real-time flows and managing energy of sensor nodes.

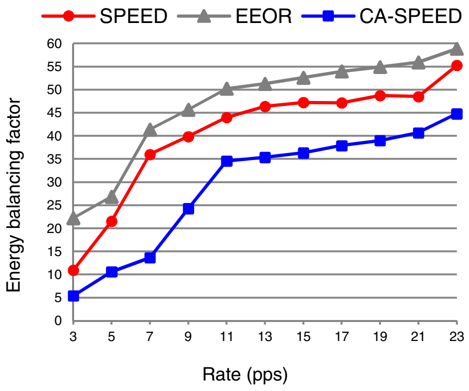
It can be clearly seen that CA-SPEED loses fewer packets (Fig. 4(a)) and consumes less energy (Fig. 4(b)) compared with EEOR and SPEED protocols. This is due to both the UPR mechanism which increases the network fluidity by removing each packet having less chance to reach its destination according to its residual



(a)



(b)

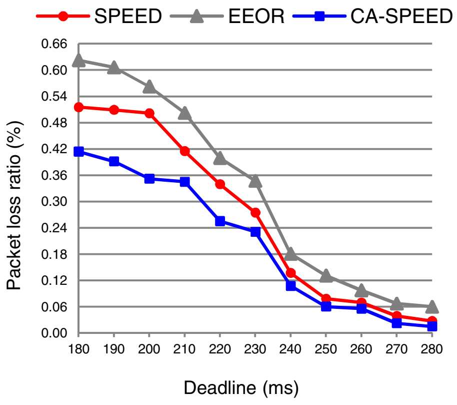


(c)

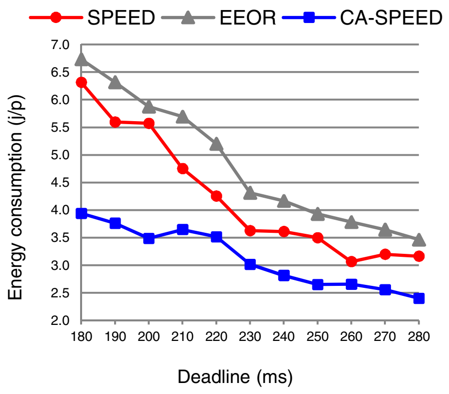
Fig. 4. Routing performance with several rates of the source nodes

deadline and the expected end-to-end delay, and the UPS mechanism which forwards first the most urgent packet extracted from the current-node queue. In applications with high rate, the EEOR and SPEED protocols lose more packets, because the deadline parameter is not considered in their routing decisions. Fig. 4(c) shows that CA-SPEED outperforms the protocols SPEED and EEOR in balancing energy of sensor nodes. This performance is due essentially to the CAB mechanism which selects as next forwarder, the neighbor with more energy when the packet is not late according to its residual deadline and expected end-to-end delay. However, it selects the neighbor providing the better relay speed when the packet is late.

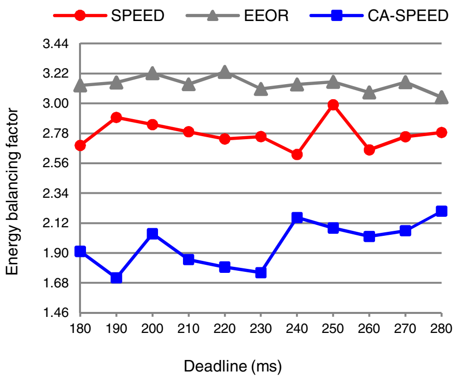
To evaluate effectiveness of our energy-aware mechanisms, used by the CA-SPEED protocol, in time critical applications, we set the rate of source nodes to 1 pps and we vary the deadline of generated packets from 180 to 280 ms. For each simulation, we measure the performance of SPEED, EEOR and CA-SPEED protocols. Obtained results, shown in Fig. 5, indicate that CA-SPEED achieves good performance compared to other protocols, especially for small packet deadlines (less than 230 ms). This is due to efficiency of the associated mechanisms (UPR, UPS and CAB).



(a)



(b)



(c)

Fig. 5. Routing performance with various data packet deadlines

Fig. 5(a) shows the positive influence of the proposed energy-aware real-time routing mechanisms on the packet loss ratio in very critical applications, particularly when the packet deadline is less than 230 ms. The CA-SPEED protocol always achieves the best performance by taking advantage of each associated mechanism (UPR, UPS and CAB). CA-SPEED achieves the lower energy consumption per delivered packet (Fig. 5(b)) because firstly, it takes advantages from the UPR mechanism to delete earlier useless packets and hence liberate bandwidth for more messages to be forwarded; secondly it forwards first the most urgent message, which, with the first mechanism, minimizes the packet loss ratio and hence increases utile consumed energy and thirdly it explicitly considers energy in its forwarding decisions using CAB. Thus, the limited energy of sensor nodes is optimally managed by the proposed mechanisms in the CA-SPEED protocol. In addition, Fig. 5(c) shows that CA-SPEED clearly outperforms the existing protocols SPEED and EEOR in node energy balancing. This is mainly due to the CAB mechanism which selects as next forwarder of the current packet the neighbor realizing the best tradeoff between the packet residual energy and the neighbor relay speed.

5 Conclusion

The work carried out in this paper deals with the the energy management problem in real-time geographic routing protocols in WSNs. To contribute in this active research field, we have proposed three energy-aware real-time routing mechanisms (UPR, UPS and CAB) that aim to improve energy managing and deliver maximum number of real-time packets in WSNs. The UPR mechanism forwards only packets with sufficient deadlines to reach their destinations. The UPS mechanism chooses the most urgent packet to be forwarded first, in order to both reduce packet loss ratio of the associated routing protocol and improve the network fluidity. This urgency is calculated relying on the expected end-to-end delay allowing the current packet to reach its destination node. The CAB mechanism provides good energy balancing while minimizing the packet loss ratio by combining both residual energy and relay speed of the forwarding candidate neighbor when selecting the next forwarder in the routing path.

We have associated the proposed mechanisms with the SPEED real-time routing protocol. The resulting protocol CA-SPEED: early removes each delayed packet in network; then selects the most urgent packet to be forwarded first; and finally forwards the selected urgent packet to the next forwarder neighbor performing the best tradeoff between the residual energy of the neighbor and its relay speed. Obtained simulation results showed that CA-SPEED outperforms the two evaluated protocols SPEED and EEOR in terms of packet loss ratio, energy consumed per delivered packet and node energy balancing.

Actually, we are developing a power-aware real-time routing mechanism which combines the adjusted transmission power of the current node with the relay speed of the forwarding candidate neighbors when selecting the next forwarder of the current packet. This mechanism will, then, be combined with the CAB mechanism in a hybrid routing approach. The resulting cost-power-aware mechanism should deliver the maximum of real-time packets, save and balance more effectively the limited energy of sensor nodes. The useless delayed packets will be removed early and the most urgent packet in the node queue will be always forwarded first.

Since we base dropping decisions concerning delayed packets simply on estimated travel times towards the sink, our future work will consider any kind of weights, urgencies, fairness, or importance values of packets in order to obtain a less aggressive approach. We also plan to put our developed source codes in Imote2 sensor nodes for experimental tests in order to consolidate the simulation results presented in this paper.

Acknowledgment. Parts of this paper have been published in: M. Aissani, S. Bouznad, B. Djamaa, I. Tsabet: "A Power-Aware Real-Time Routing Mechanism for Wireless Sensor Networks", 9th International Conference on Networking and Services (ICNS), 2013.

References

[1] Akkaya, K., Younis, M.: A Survey on Routing Protocols for Wireless Sensor Networks. Ad Hoc Networks 3(3), 325–349 (2005)

[2] Ehsan, S., Hamdaoui, B.: A Survey on Energy-Efficient Routing Techniques with QoS Assurances for Wireless Multimedia Sensor Networks. IEEE Communications Surveys & Tutorials 14(2), 265–278 (2012)

[3] Marjan, R., Behnam, D., Kamalrulnizam, A.B., Malrey, L.: Multipath Routing in Wireless Sensor Networks: Survey and Research Challenges. Sensors Journal 12(1), 650–685 (2012)

[4] Rezayat, P., Mahdavi, M., Ghasemzadeh, M., AghaSarram, M.: A Novel Real-Time Power Aware Routing Protocol in Wireless Sensor Networks. Journal of Computer Science 10(4), 300–305 (2010)

[5] Yang, W., Liang, W., Dou, W.: Energy-Aware Real-Time Opportunistic Routing for Wireless Ad Hoc Networks. In: Proceedings of the IEEE Global Telecommunications Conference, Miami, FL, USA, pp. 1–6 (2010)

[6] Mao, X., Tang, S., Xu, X.: Energy-Efficient Opportunistic Routing in Wireless Sensor Networks. IEEE Transactions on Parallel and Distributed Systems 22(11), 1934–1942 (2011)

[7] Xue, L., Guan, X., Liu, Z., Yang, B.: TREE: Routing strategy with guarantee of QoS for industrial WSNs. International Journal of Communication Systems (IJCS), `http://onlinelibrary.wiley.com/doi/10.1002/dac.2376/full` (last accessed June 20, 2012)

[8] He, T., Stankovic, J.A., Lu, C., Abdelzaher, T.: A Spatiotemporal Communication Protocol for Wireless Sensor Networks. IEEE Transactions on Parallel and Distributed Systems 16(10), 995–1006 (2005)

[9] Zollinger, A.: Networking unleashed: Geographic routing and topology control in ad hoc and sensor networks. PhD thesis, ETH Zurich, Switzerland, Diss. ETH 16025 (2005)

[10] Soro, S., Heinzelman, W.: A Survey of Visual Sensor Networks. Advances in Multimedia 21, 1–21 (2009)

[11] Liu, K., Abu-Ghazaleh, N., Kang, K.D.: JiTS: Just-in-Time Scheduling for Real-Time Sensor Data Dissemination. In: Proceedings of the 4th IEEE Annual International Conference on Pervasive Computing and Communications, Pisa, Italy, pp. 42–46 (2006)

[12] Levis, P., Gay, D.: TinyOS programming. Cambridge University Press, USA (2009)

[13] Levis, P., Lee, N., Welsh, M., Culler, D.: TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In: Proceedings of the 1st ACM Conference on Embedded Networked Sensor Systems, LA, California, USA, pp. 126–137 (2003)