Advance Access publication on April 7, 2010
doi:10.1093/comjnl/bxq038

A Hybrid Adaptive Protocol for Reliable Data Delivery in WSNs with Multiple Mobile Sinks

GIUSEPPE ANASTASI[1], ELEONORA BORGIA[2,*], MARCO CONTI[2] AND ENRICO GREGORI[2]

[1]*Department of Information Engineering, University of Pisa, via Diotisalvi 2, 56122 Pisa, Italy*
[2]*IIT-CNR, via G. Moruzzi 1, 56124 Pisa, Italy*
Corresponding author: eleonora.borgia@iit.cnr.it

In this paper, we deal with reliable and energy-efficient data delivery in sparse wireless sensor networks (WSNs) with multiple mobile sinks (MSs). This is a critical task, especially when MSs move randomly, as interactions with sensor nodes are unpredictable, typically of short duration and affected by message losses. In addition, multiple MSs can be simultaneously present in the sensor contact area making the minimum energy data delivery a complex optimization problem. To solve the above issues, in this paper we propose a novel protocol that efficiently combines erasure coding with an Automatic Repeat reQuest (ARQ) scheme. The key features of the proposed protocol are as follows: (i) the use of redundancy to cope efficiently with message losses in the multicast environment and (ii) the ability of adapting the level of redundancy based on feedbacks sent back by MSs through ACKS. We observed by simulation that our protocol outperforms an alternative protocol that relies only on an ARQ scheme, even when there is a single MS. We also validated our simulation results through a set of experimental measurements based on real sensor nodes. Our results show that the adoption of encoding techniques increases the lifetime of the sensor in the range (40–55%) compared with standard simple ARQ approaches when applied to WSNs with MSs.

Keywords: wireless sensor networks; mobile sinks; reliable data delivery; erasure coding; ARQ

Received 8 March 2010; revised 8 March 2010
Handling editor: Erol Gelenbe

1. INTRODUCTION

Wireless sensor networks (WSNs) generally consists of a large number of tiny sensor nodes densely deployed over an area to collect physical information from the surrounding environment [1]. However, many common WSN applications do not require fine grain sensing and, hence, sensor nodes can be placed very far from each other, forming *sparse sensor networks*, i.e. networks where the distance between neighboring nodes is larger than their transmission range. In sparse WSNs, data collection can be accomplished through (special) mobile nodes moving in the environment that opportunistically gather information from sensor nodes when they are within the nodes' transmission range. In this paper, we refer to applications where data produced by sensor nodes are gathered and directly consumed by mobile nodes, which therefore can be referred to as mobile sinks (MSs). For example, this scenario could correspond to the case of sensors located in an urban environment (e.g. along streets, at traffic lights, at bus stops) and measuring air quality parameters, meteorological data or other information relevant to citizens and/or visitors. Data produced by sensor nodes are gathered and consumed by people moving in the city using their personal devices.

In this scenario, data delivery to MSs can occur only during contact times, i.e. the sensor broadcasts its collected data to all the MSs in its communication range. In addition, since MS arrivals are generally not known in advance, the data delivery process requires a continuous discovery phase performed by the sensor node to detect MSs within its contact area. Both the discovery and data delivery phases should be energy efficient as energy is the most critical resource in a sensor node.

An important factor affecting the data delivery process is the contact duration. The actual contact time mainly depends on the path followed by the specific MS and its speed. However, it is typically short and its duration is generally unknown to the static sensor. Hence, a reliable and efficient broadcast communication protocol, capable of transmitting a large amount of data in a short contact time with minimal energy expenditure at the sensor node, is required for data delivery.

Another factor that impacts on the energy consumed by the sensor for data delivery is that, when different MSs are in contact at the same time, they typically have entered the contact area at different time instants. For example, let us assume that the static sensor has discovered an MS at a generic time instant t_0. When a second MS is discovered, at a time instant $t_1 > t_0$, the first MS may have already received a large amount of data. The same occurs when a third MS is discovered at a time $t_2 > t_1 > t_0$, and so on. Hence, the broadcast delivery protocol must be capable of managing efficiently parallel delivery processes—with different MSs—which are only partially overlapped.

The major contribution of this paper is the design of a novel, energy-efficient communication protocol for this context, named *Hybrid Interleaved* (HI) *data delivery*. HI provides reliable and energy-efficient data transfer from a sensor node to all MSs within its transmission range by efficiently combining an encoding technique (ET) (i.e. Reed-Solomon (RS) codes [2]) with an Automatic Repeat reQuest (ARQ) scheme in an adaptive way. The proposed protocol has been evaluated through an extensive simulation analysis. The obtained results have shown that our hybrid adaptive protocol largely outperforms another protocol based on a traditional Selective Retransmission scheme, even when there is a single MS.

Another major contribution of the paper is the validation carried out with real sensor nodes. The computational burden caused by RS-codes in the sensor node is almost impossible to be quantified in simulation and, hence, the only way to get ground truth figures is to set up experiments using real sensor nodes. The experimental results have shown that (i) current sensors have enough computational power to manage the RS-coding process and (ii) in terms of energy consumption, coding is typically negligible with respect to the data transmission.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 introduces the design principles followed in the protocol definition. Section 4 describes the HI protocol. Section 5 presents the simulation setup used for our performance analysis. The simulation results are discussed in Section 6 and validated in Section 7 through a set of experimental measurements. Finally, Section 8 concludes the paper.

2. RELATED WORK

The bibliography on WSNs with mobile data collectors (MDCs) (i.e. MSs or mobile relays[1]) is extremely large. In this section, we will focus on protocols for reliable and energy-efficient data exchange between a static sensor and the MDC. A more general discussion on sensor networks with MDCs can be found in [3, 4].

The idea of using MDCs was first proposed independently in [5, 6] to address the problem of energy-efficient data collection in sparse sensor networks. Then it was shown that using mobile nodes for data collection can be beneficial also in dense sensor networks [7]. Data collection/dissemination through mobile elements has been considered also in the context of ad hoc networks [8].

In [9], the MDC-based approach is evaluated by means of analysis and simulation. The authors investigate the impact of a large set of operating parameters on the data success rate, latency and energy consumption. They assume an ideal channel and no specific data-transfer protocol and, hence, the probability of data reception is mostly affected by buffering constraints. In [10], the authors investigate the use of multiple MDCs for data collection, since a single MDC cannot be sufficient in some environments. They consider techniques to balance the number of static sensor nodes served by an MDC. They assume *coordinated* MDCs and primarily study *load balancing*. Our goal is to maximize the (energy) *efficiency* of the data transfer phase. Another major difference is that we assume *uncoordinated independent* MDCs that may happen to be simultaneously in the contact area of the same sensor node.

Reliability in data transfer from the sensor node to the MDC is typically achieved through an ARQ scheme. Acknowledgement-based data-transfer protocols are considered, for example, in [7, 11–16] to tackle with both channel errors and possible collisions. Some of these works [7, 11, 15, 16] assume that MDC mobility can be controlled in order to extend network lifetime, improve reliability of data communication and reduce resource consumption and latency. Therefore, such approaches usually assume that contact times between the MDC and a sensor node are long enough to successfully complete the data transfer. In this paper, this assumption is relaxed, i.e. no specific assumption is made about MDC mobility, duration of contact times and message loss pattern. As a result, our proposed data-transfer protocol is very general.

Data-transfer protocols based on ETs [17] have been extensively used for reliable data transfer in multi-hop ad hoc networks, including traditional (i.e. static multi-hop) sensor networks [18–21] and underwater sensor networks [22]. Specifically, network coding has shown to be a very promising solution for data dissemination in multi-hop ad hoc networks as it is able to provide very high reliability and exploits bandwidth very efficiently [23]. Attention has also been devoted to possible applications of ETs for data dissemination in mobile ad hoc networks [24, 25], where end-to-end connectivity is not guaranteed, and communication between neighboring nodes occurs only when they happen to meet each other. In [24], the authors propose a forwarding scheme—based on network coding—for efficient delivery of messages and in [25] a similar approach is taken but with the use of

---

[1] Mobile relays are special nodes that collect and transport data to a data collection point for further processing.

rateless codes, instead of network coding. Both works refer to scenarios with multi-hop unicast communications and exploit data redundancy to increase the delivery probability of each single message to the final destination (which is not guaranteed due to intermittent connectivity between nodes). In this paper, we refer to *bundle-oriented* applications, where a number of messages have to be reliably delivered to the destination, and focus on single-hop communication. In addition, we consider both unicast (i.e. single MDC) and multicast (i.e. multiple MDCs) communications.

The idea of using ETs for reliable multicast communication has been already exploited in traditional networks [26]. In this paper, we show that such an approach can be effectively used also in sensor networks with MDCs, and that it is appropriate not only for multicast communications, as one would expect, but also for unicast communications (i.e. when there is a single MDC).

3. DESIGN PRINCIPLES

In this paper, we focus on a specific class of WSN applications, throughout referred to as *bundle-oriented* applications. In such applications, the static sensor node has a limited amount of data (e.g. measurements of air pollution level in the last hours, or days) to be delivered, on demand, to mobile users that happen to be within its contact area. The data transfer is accomplished through a *bundle* of consecutive messages sent by the static sensor to the mobile user. The mobile user then consumes data for its own purposes, thus acting as an MS. The sensor network is assumed to be sparse and, hence, at a given time each MS is in contact with at most one static node. Instead, several MSs can be simultaneously within the contact area of the same sensor node. As shown in Fig. 1, the various MSs will experience different contact times and link qualities, depending on how their path crosses the contact area. Static sensors are resource constrained, energy being the most critical one, whereas MSs are assumed to have large computational resources and no energy limitation (as their battery can be replenished). This scenario fits the case of sensors deployed in an urban environment (e.g. along streets, at traffic lights, at bus stops) and MSs represented by people walking or in cars moving around the city and using their personal devices (e.g. PDAs, smartphones).

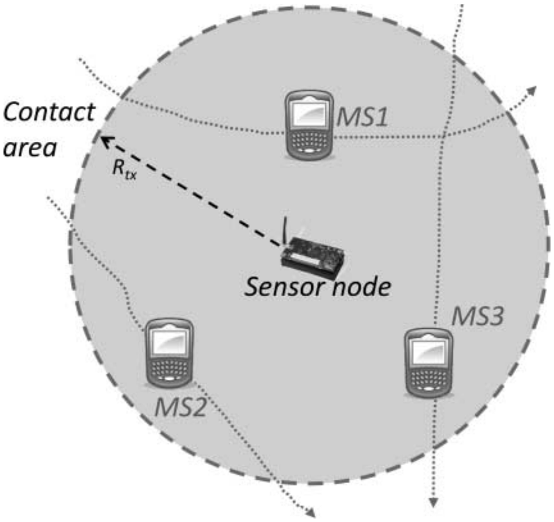

FIGURE 1. Reference model.

In the reference scenario introduced above, the contact time is a limited and scarce *resource* that should be exploited very efficiently by the communication protocol used for delivering messages to MSs. Contact times are *very short* if the MS moves fast and/or the sensor node operates with a low duty cycle to save energy, and *scarce* because contact times occur rarely and the communication may experience severe message losses.

In this context, the goal of the communication protocol should be transmitting all the available data during the contact time with minimal energy consumption at the static sensor node.

As highlighted in [27], chatty communication paradigms are not suitable for environments where contact durations are not predictable or are expected to be short. Instead, communication protocols with minimum interaction between the sensor node and MSs are preferable. In this perspective, a valuable strategy is making use of ETs [17, 28]. Basically, when ETs are applied to networking protocols, data is not sent plain but combined (encoded) into blocks of data. A source node willing to send k messages encodes these k messages into n encoded messages, with $n >> k$. A receiving node does not need to receive exactly the k original messages: any set of n out of the n encoded messages generated at the source is sufficient to decode the k original messages. This property improves system robustness against data losses.

One of the major issues concerning ETs is the computational burden involved in both the encoding and decoding processes. However, previous work has demonstrated that software implementations are feasible also for obsolete, low-performing architectures [29], as well as small, resource-constrained devices [18, 19].

Another drawback is connected to the redundancy level to be introduced. In fact, when using Erasure Codes (i.e. a particular ET scheme), the redundancy level is fixed at the beginning and controlled by the *stretch factor* (i.e. n/k). This guarantees a fixed degree of loss tolerance: a receiver can recover from up to $n-k$ losses in a group of n encoded blocks. Tuning of the stretch factor has a huge impact on the protocol performance, but it is very difficult to carry out if more MSs are within the contact area and willing to gather the same data (i.e. this scenario is similar to the multicast case). If the stretch factor is set to a low value, far MSs experiencing a high message loss might not receive a sufficient amount of information to complete the decoding process, since low redundancy is introduced. On the contrary, if MSs are close to the sensor, a high stretch factor causes resource wastage since the sensor transmits all the n codes but some of them are not used by the decoder. An improvement might be obtained, for example, adapting the stretch factor to the varying message loss during the contact time. However, this is a very hard task since MSs enter the contact area at different times and, thus, they experience different message loss patterns (typically the message loss probability is high at the beginning and at the end of the contact, and low when the MS is very close to the sensor [14]).

For this reason in the HI data delivery protocol we follow an alternative approach: we create in advance enough redundancy (high stretch factor) but we choose dynamically the number of codes to be transmitted using feedbacks sent by the MS(s). Hence, HI is a *hybrid* protocol since it combines an ET-based approach with an ARQ scheme. Specifically, for the ET component of the protocol we use the *RS-codes* (see Appendix and [2] for details). Several types of erasure codes (e.g. Rateless codes [30, 31]) have been proposed, however, most of them are not optimized for systems with low computational capabilities such as WSNs. The main drawback of using RS-codes is the encoding phase which has a quadratic order in contrast with the linear one used by other erasure coding techniques. However, this does not affect the system performance since in our protocol the redundancy is produced in advance, as it will be discussed below. Instead, the main strength of RS-codes is the decoding phase which is faster than the other approaches since the destination requires the minimum number of codes to be able to reconstruct the original data. As a consequence we will focus on RS-codes as an efficient representation of Erasure Codes.

To sum up, the basic idea is to produce enough redundancy in advance, and send codes on demand, depending on feedbacks sent back by MSs. In this way, the encoding process at the sensor node is performed just once and this allows to optimally use the contact times. In addition, the protocol is *flexible* thanks to its ability to adapt the number of codes to be transmitted based on feedbacks sent back by each MS (i.e. number of messages still required to complete the decoding process).

4. PROTOCOL DESCRIPTION

Before giving a detailed description of the HI protocol, it may be worthwhile pointing out the assumptions it relies upon.

(i) Contacts between the sensor node and MSs occur randomly, i.e. visit times of MSs cannot be predicted in advance by the sensor node. Therefore, the sensor node must be in a discovery state—typically with a low duty cycle to save energy—while waiting for MSs.

(ii) To announce its presence, an MS periodically broadcasts *beacon* messages. Upon receiving a beacon, the sensor realizes that a contact with an MS has been established. Hence, it switches to a 100% duty cycle, and starts the data delivery process.

(iii) Contact times have unpredictable duration. Therefore, the sensor node relies on ACKS received during the data delivery to infer about the presence of the MS. Specifically, after missing a predefined number NACK of consecutive ACKS, the sensor node assumes that the contact has been lost. This avoids sending data uselessly when the MS is too far away.

Obviously, the performance of the HI protocol is strongly influenced by the parameters used in the discovery phase, i.e. the beacon period (T_B) and the sensor's duty cycle (D). For example, a high duty cycle allows an earlier discovery of the approaching MS—thus ensuring a longer residual contact time—but consumes more energy.

4.1. Protocol operations

As mentioned in Section 3, we assume that a data bundle of limited size has to be sent to one or more MSs that happen to be within the contact area of the sensor node. Figure 2 depicts the operations required to transfer the bundle to an MS. The original bundle (i.e. source data units in Fig. 2) is first encoded by the source node into a wider bundle of encoded data units utilizing the RS-coding scheme (see Appendix). Encoded data units are then transmitted to the MS through the HI protocol. At the destination side, encoded data units are decoded to reconstruct the original data units (Fig. 2). The RS-coding implemented in HI follows the approach suggested in [26]. Before encoding, the entire bundle is subdivided into B blocks (i.e. $B_0, B_1, \ldots, B_{B-1}$), with each block consisting of k data units (Fig. 3a). Each block is then encoded separately. Each encoded block contains n encoded data units: assuming that systematic codes are used, the first k data units are equal to the original data units and the additional $n - k$ are redundant encoded data units (Fig. 3b). The source node schedules for transmission encoded data units picked from consecutive blocks rather than sequentially chosen from the same block, as shown by arrows in Fig. 3b (i.e. interleaved scheme). This interleaved scheme guarantees that messages losses are uniformly distributed among all blocks, rather than concentrated in a single block. Obviously, we assume that both the sensor node and the MS(s) are aware of the encoding parameters, that is, the number of original messages (k) and blocks (B) within a bundle, and the encoding matrix.

Upon discovering at least one MS, the sensor node starts to transmit encoded messages using the interleaved scheme described above. Each encoded message contains: (i) the *block identifier* (i.e. $0, 1, \ldots, B-1$), (ii) the *sequence number* within the block (i.e. $1, 2, \ldots, n$) and (iii) the *encoded data unit*. The first two information are essential for the MS to understand when it has received a sufficient number of messages to decode the original bundle (i.e. using the interleaved scheme it has to receive at least k different messages for each block to decode

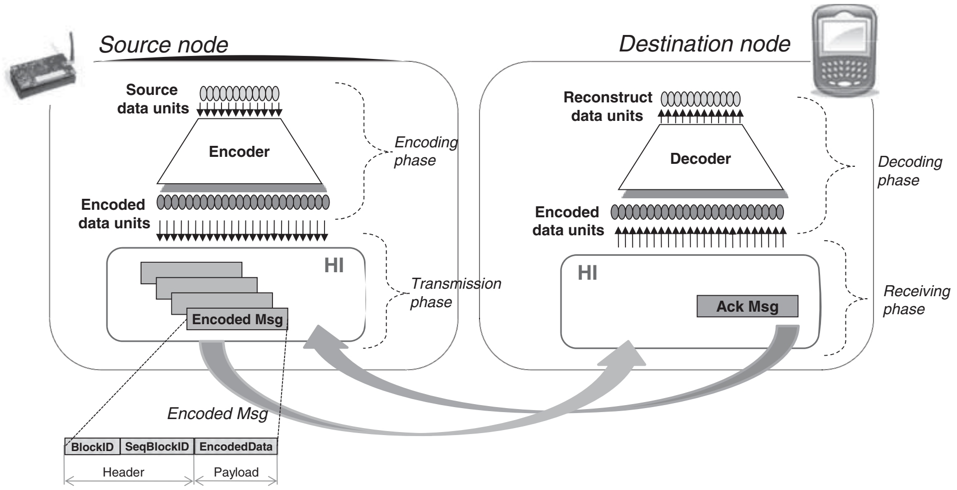

FIGURE 2. Overview of the system architecture.

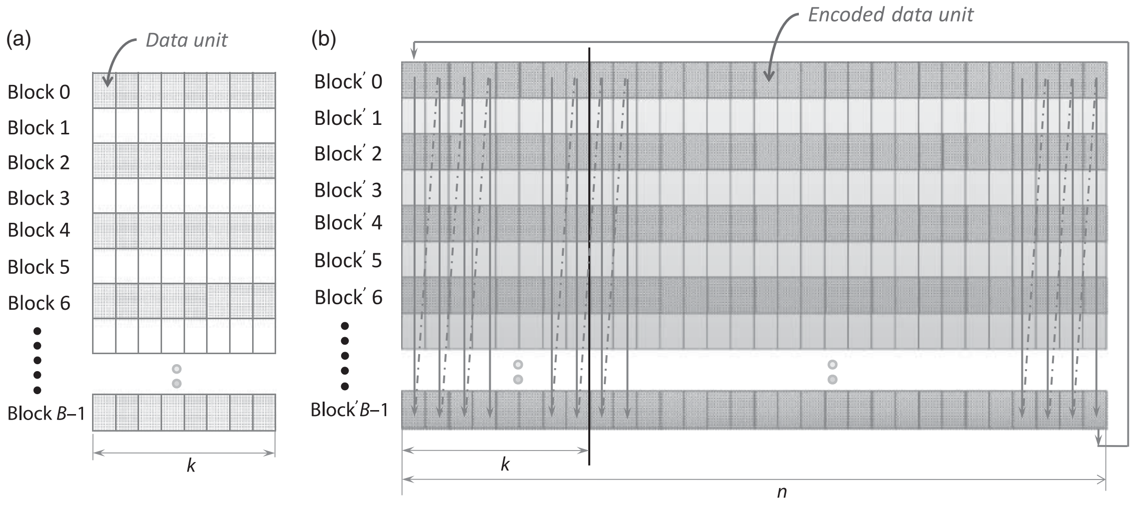

FIGURE 3. The original bundle (a) and the encoded bundle (b).

it). The MS uses this information to generate Ack messages. They are sent periodically by the MS (every TACK) and notify, for each block, how many different encoded messages have been correctly received by the MS through a mask (i.e. MaskBlockID field). The sensor node collects all the incoming ACKs and stores, for each block, the lowest received value.

When one or more block values are lower than k, which corresponds to the existence of one or more MSs requiring additional encoded messages to decode the bundle, additional data transmissions are needed. Thus, the sensor continues transmitting encoded messages, starting from the last message sent, using the interleaved scheme but skipping those blocks already completed by all the MSs (if any). This guarantees the transmission of only useful encoded messages. The process goes on until the minimum set of encoded messages has been received by all the MSs (i.e. all the block values stored at the sensor node are equal to k), or all MSs are out of the contact area. Hence, the protocol is able to adapt to different levels of message losses experienced by different MSs. It is worth emphasizing here that ACKs introduce a very limited overhead as they also serve as implicit beacons.

An example of the bundle transmission protocol described above is presented in Fig. 4 assuming one sensor and two MSs. The figure highlights the following: (i) the loss of encoded messages and/or ACK messages have a limited impact on the overall performance and (ii) MS arrival times and bundle decoding times are asynchronous events.

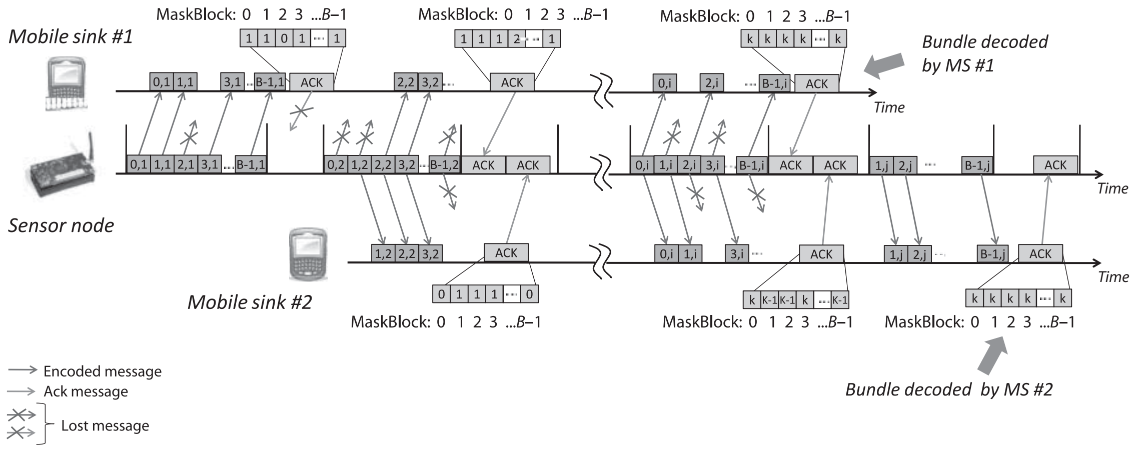

FIGURE 4. Example of bundle transmission protocol.

5. SIMULATION SETUP

To evaluate the performance of the proposed protocol, we implemented the HI protocol in an event-driven simulator designed and implemented from scratch. For comparison we also considered and implemented an ARQ-based protocol—that uses the traditional *selective repeat* (SR) [32] scheme for recovering lost messages—to compare the performance of these two approaches. This protocol, throughout referred to as the SR protocol, is briefly described in the following section.

5.1. SR protocol

As the name suggests, the SR [32] protocol avoids unnecessary retransmissions on the basis of a mechanism in which (i) the sensor node transmits bursts of data messages sequentially, (ii) each receiver individually acknowledges the messages received correctly and (iii) the sensor node retransmits messages not acknowledged. For the sake of clarity, we first describe the protocol operations when there is a single MS, and then we extend the description to the case of multiple MSs.

Upon discovering the presence of one MS in the contact area, the sensor node starts the transmission of the bundle[2]. In this case, the bundle is divided into N messages which are labeled from 0 to $N - 1$. Messages are transmitted in bursts following the sequential order (starting from 0) and wrapping around upon reaching the end of the bundle, if necessary. The MS receives and stores messages in its local buffer and then sends back the acknowledgment. ACKS contain the sequence number of the last message received in order and a bit mask indicating which messages the MS has (has not) received correctly. Upon receiving an ACK, the sensor node retransmits all missed messages, starting from the last acknowledged one. Then, it continues transmitting new messages until the MS has received all the N messages of the bundle or it has moved out of the contact area.

In the case of multiple MSs joining the transmission at different instants, the sensor node gives priority to the MS that has the best channel condition. For this reason a counter, namely $\text{NACK}_{\text{MS}_i}$, takes into account the number of lost ACKS sent by MS_i. $\text{NACK}_{\text{MS}_i}$ is initially equal to 0, increases each time the sensor does not receive an ACK sent by MS_i and it is reset when the sensor receives it correctly. The sensor uses this information to choose the MS for retransmission: it gives the priority to the MS that has the lowest NACK_{MS} value, i.e. the MS that has the best channel condition. If two or more MSs have the same lowest value, it selects the first MS entered in the contact area. We observed by simulation that this optimized strategy increases the probability of completing the bundle delivery.

5.2. Performance metrics

The performance comparison between the HI and SR protocols is based on the following performance metrics:

(i) *Decoding Probability*: probability of receiving the minimum amount of bytes for an MS being able to decode the original data bundle (in the SR protocol, the probability of receiving the complete bundle).

(ii) *Decoding Latency*: time interval between the reception of the first message by MS and the successful decoding of the bundle (in the SR protocol, the time interval required to receive the entire bundle). This index is computed only on those MSs that have correctly decoded the bundle.

(iii) *Energy:* average total energy consumed by the sensor node per each byte correctly transferred to the MS. It

---

[2] Note that the detection of MSs entering and exiting the contact area is performed in the same way as in the HI protocol.

can be calculated as

$$\text{Energy} = \frac{(m \cdot \delta_{\text{MSG}} \cdot P_{\text{tx}}) + (m \cdot \delta_{\text{MSG}}/T_{\text{ACK}}) \cdot N_{\text{MS}}(\delta_{\text{ACK}} \cdot P_{\text{rx}})}{B_{\text{tot}}}$$

where m is total number of messages transmitted by the sensor node; $\delta_{\text{MSG}}(\delta_{\text{ACK}})$ represents the time required to transmit a message (ACK); $P_{\text{tx}}(P_{\text{rx}})$ indicates the power consumed by the sensor node in the transmit (receive) mode; T_{ACK} is the time interval between two consecutive ACKS sent by the same MS; N_{MS} is the number of MSs considered in the experiment; and finally, B_{tot} is the total number of bytes decoded by all the MSs.

In the expression above, the numerator represents the total energy consumed by the sensor node. Specifically, $m \cdot \delta_{\text{MSG}} \cdot P_{\text{tx}}$ measures the total energy consumed for transmitting all data messages, while the second addendum at the numerator accounts for the energy spent for receiving ACKS from all the MSs ($(m{\cdot}\delta_{\text{MSG}}/T_{\text{ACK}}){\cdot}N_{\text{MS}}$ gives the total number of ACKS received by the sensor node). The denominator is the total number of bytes decoded by all the MSs.

(iv) *Goodput*: ratio between the number of useful bytes and the total number of bytes received by the MS.

5.3. Simulation parameters

In our simulation analysis, we study the scenario with a single static sensor node and one or more MSs (depending on the experiment); this scenario is motivated by the sparse network assumption. We also assume that MSs move along a linear path at a fixed (vertical) distance from the sensor node, at a constant speed. The sequence with which MSs join the contact area is the following: assuming that the first MS_0 enters at the generic instant time t_0 and has a contact time $c_{\max}$, the second one (MS_1) enters at a random time t_1 uniformly distributed in the interval $[t_0, t_0 + c_{\max}]$, the third one (MS_2) enters at a random time t_2 uniformly distributed in the interval $[t_1, t_0 + c_{\max}]$, etc.

We consider three mobility scenarios characterized by different speeds for MSs. In the *High Mobility* scenario MSs are assumed to be on board of buses or cars in a typical urban environment. Therefore, the considered speed is 40 km/h. On the contrary, in the *Low Mobility* scenario MSs are assumed to be personal devices carried by pedestrians. Thus, we consider a speed of 3.6 km/h. Finally, in the third scenario, referred to as *Heterogeneous Mobility* scenario, we assume a heterogeneous environment, where MSs are carried by cars or pedestrians. In this scenario, we consider two speeds for a car, i.e. 40 and 20 km/h.

In all three scenarios, message (ACK) loss probability is computed by using the model considered in [14], and derived from experimental measurements taken in a scenario similar to the one considered here [33]. Specifically, we use a polynomial message loss probability function in the form

$$p(t) = a_2 \left(t - \frac{c_{\max}}{2}\right)^2 + a_1 \left(t - \frac{c_{\max}}{2}\right) + a_0, \qquad (1)$$

where t represents the time elapsed since the initial contact and $c_{\max}$ represents the nominal contact time. Equation (1) holds only within the contact area. Outside of the contact area the message loss probability is assumed to be equal to one (i.e. any transmitted message is lost). Note that $p(t)$ does not take into account losses due to collisions, but only due to transmission errors. In our environment, we have one sensor and several MSs. Collisions can occur when two or more MSs want to transmit an ACK at the same time. In our simulator before transmitting ACKS, MSs wait for a random time and if two or more MSs choose the same time instant for transmission, the ACKS are lost due to collision. Now $p(t)$ is applied to those packets that are not lost due to collisions.

To derive the coefficients in Equation (1)—reported in Table 1 for different speeds and for a vertical distance from the sensor node equal to 15 m—we used the same methodology described in [14]. Briefly, a polynomial interpolation of real probability loss measured in [33] has been derived by using the least square interpolation method. To decide the degree of the polynomial function and the corresponding coefficients, the performance of a very basic data-transfer protocol has been compared when using the real packet loss curve and the polynomial function. Such analysis has been demonstrated that a 2-degree polynomial function is sufficiently accurate. Figure 5 shows how the polynomial loss function approximates the real packet loss experienced by an MS that is moving at 3.6 km/h.

For each considered scenario we performed several sets of experiments, characterized by different number of MSs and bundle sizes. Table 2 shows the values used for fixed parameters. Each experiment consists in sending a bundle of messages from the sensor node to the MS(s).

To derive confidence intervals, we used the independent replication method with a 90% confidence level. In all experiments we performed 10 replicas, each consisting of 10 000 contact times.

TABLE 1. Message loss parameters for the low, high mobility and heterogeneous scenario.

Parameter	$v = 3.6$ km/h	$v = 20$ km/h	$v = 40$ km/h
$c_{\max}$	158 s	30 s	17 s
a_0	0.133	0.3828	0.4492
a_1	$0\ \text{s}^{-1}$	$0\ \text{s}^{-1}$	$0\ \text{s}^{-1}$
a_2	$0.000138\ \text{s}^{-2}$	$0.0028\ \text{s}^{-2}$	$0.0077\ \text{s}^{-2}$

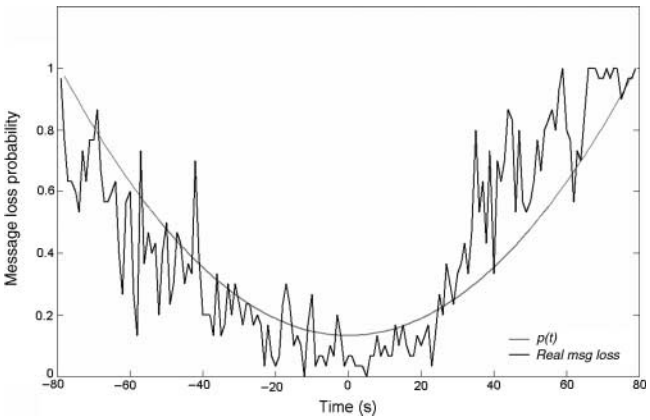

FIGURE 5. Probability loss function $p(t)$ derived for 3.6 km/h.

TABLE 2. Simulation parameter setting.

Parameter	Value
k, n (HI protocol)	8, 256
Message/ACK size	110 bytes
Message transmission time (δ_{MSG})	17 ms
ACK transmission time (δ_{ACK})	17 ms
ACK period (T_{ACK})	$16 * \delta_{\text{ACK}}$
Beacon period (T_{B})	100 ms
NACK (40, 3.6 Km/h)	8, 24
Duty cycle (D)	5%
Transmission power (P_{tx})	52.2 mW
Reception power (P_{rx})	56.4 mW

6. SIMULATION RESULTS

In this section, we compare the performance of the HI and SR protocols in the three mobility scenarios introduced above. Clearly, the performance of HI depends on the level of redundancy used by the sensor node in transferring messages to the MS(s) as, intuitively, a higher redundancy level allows a better decoding probability at the cost of an increased energy consumption at the sensor node. Therefore, we performed a set of preliminary experiments to determine the most appropriate redundancy level to be used in the subsequent analysis. The results of this preliminary analysis are discussed in the next section.

6.1. Impact of redundancy

Assuming that k is the number of original messages in each block[3], we define the following four redundancy levels:

(i) *Level 0*: no redundant code is generated, i.e. $n = k$[4];
(ii) *Level 1*: a number of redundancy codes equal to the number of original messages are generated, i.e. $n = 2k$;
(iii) *Level 2*: an intermediate number of redundant codes are generated;
(iv) *Level 3*: the maximum number of codes is generated (this corresponds to $n = 2^k$ codes, in order to operate on Extension Galois Field).

It may be worthwhile recalling here that the generated redundant codes are not necessarily transmitted. The sensor node sends only the minimum number of redundant codes that allow one to decode the bundle at the MS (see Section 4).

In our analysis, we considered a medium size bundle consisting of 14 080 bytes subdivided into 16 blocks of 8 messages. Accordingly, the values for $n - k$ and n when using the different redundancy levels are shown in Table 3. Figure 6 shows the decoding probability and energy consumption for four redundancy levels and up to 10 MSs in the *High Mobility* scenario[5] (the results in the other scenarios are similar and are, thus, omitted). Note that the x-axis represents the maximum number of MSs which can be simultaneously in contact with the sensor node. As expected, for a fixed number of MSs, the decoding probability increases with the redundancy level. This is because a larger number of available codes increases the probability of sending fresh and, thus, useful information during the contact time. Correspondingly, the energy consumed per byte correctly decoded by the MS decreases when the redundancy level increases, i.e. the protocol tends to become more energy efficient. The reason behind this is that a greater decoding probability implies a more efficient utilization of the energy consumed by the sensor node to transmit messages. Figure 6 highlights that there is a large increase in performance when passing from Level 0 (no redundancy) to Level 1 (number of redundancy codes equal to the number of original messages). Increasing the degree of redundancy beyond Level 1 still provides some improvement in terms of the decoding probability. Beyond Level 2 there is no significant effect. Figure 6 also shows that, as expected, the benefit of using redundancy is higher for a large number of MSs.

Since redundant codes are generated in advance (i.e. the generation process does not interfere with the transmission process) and only the minimum number of codes is actually transmitted, in the following experiments we will consider the maximum redundancy level (i.e. Level 3). This allows us to better understand the potentials of the HI protocol. However, based on the previous results, in a real implementation a lower

---

[3] Note that the total bundle size (measured in messages) is equal to $k \cdot B$, where B is the number of blocks of the bundle.

[4] This case is similar to the SR protocol since only the original messages are sent. The difference is related to the way they manage retransmissions.

[5] Having five or more MSs near a sink is realistic in the urban environment we have envisaged. This could be the case of a sensor that distributes popular information (e.g. traffic information, advertisements) and is located in a strategic position (e.g. traffic light, bus stop).

TABLE 3. Redundancy levels considered in the preliminary analysis.

Redundancy level	$n - k$	n
Level 0	0	8
Level 1	8	16
Level 2	24	32
Level 3	248	256

redundancy level may be a better option, especially when sensor nodes have limited CPU and/or memory capabilities. We will further discuss this issue in Section 8.

6.2. High mobility scenario

We start our analysis by considering the *High Mobility* scenario. This is a critical scenario due to the speed of MSs (40 km/h) which limits the duration of the time interval available for receiving messages from the sensor node. The nominal contact time is ∼17 s in this scenario, but note that a fraction of this time is needed to discover the MS.

Figure 7 shows the performance metrics for several bundle sizes and number of MSs (in this and subsequent figures dashed and solid lines refer to HI and SR, respectively). We first analyze the case of a single MS (square marker) in the contact area. Intuitively, one would expect that the SR protocol outperforms the HI protocol in this specific case, as HI introduces redundancy proactively, while SR retransmits only missed messages. However, the results in Fig. 7 show that the two protocols exhibit very similar behavior in this specific case (curves are almost overlapped), and HI tends to outperform SR for short bundle sizes. These results can be explained by taking into account that the MS needs to receive k *independent* messages for each block of data composing the bundle when using HI, while it must receive *all* (k) messages in each block when using SR. When the bundle size is small, in the SR protocol the sensor node may transmit all messages in the bundle before receiving an ACK from the MS (ACKs may get lost). Upon reaching the end of the bundle, the sensor node starts retransmitting messages from the beginning. Hence, the MS may receive duplicate messages that are useless and consume energy. On the contrary, in the HI protocol the sensor node always transmits independent codes that can be used by the MS.

As expected, HI largely outperforms SR with respect to all considered performance indexes when the number of MSs, within the contact area, is larger than one. This is because, in the HI protocol, redundant codes sent by the sensor node can potentially be exploited by *all* MSs whereas, in the SR protocol, missed messages must be retransmitted on an *individual* basis. This aspect is better highlighted in Fig. 8, which compares the decoding probability and energy efficiency for an increasing number of MSs and three different bundle sizes (corresponding to 80, 160 and 240 110-byte messages, respectively). In general, increasing the number of MSs has two contrasting effects on the performance of both protocols. On the one hand, a larger number of MSs reduces the amount of bandwidth available for data transfer (there are more acknowledgements and, potentially, more collisions). On the other hand, when there are more MSs, the same message can be potentially used by all MSs. This is the main reason behind the increasing (decreasing) behavior of the decoding probability (energy efficiency) with the number of MSs for short bundle sizes.

6.3. Low mobility scenario

In this section, we investigate the performance in the *Low Mobility* scenario. Here MSs are supposed to be carried by pedestrians (e.g. MSs could be personal devices used

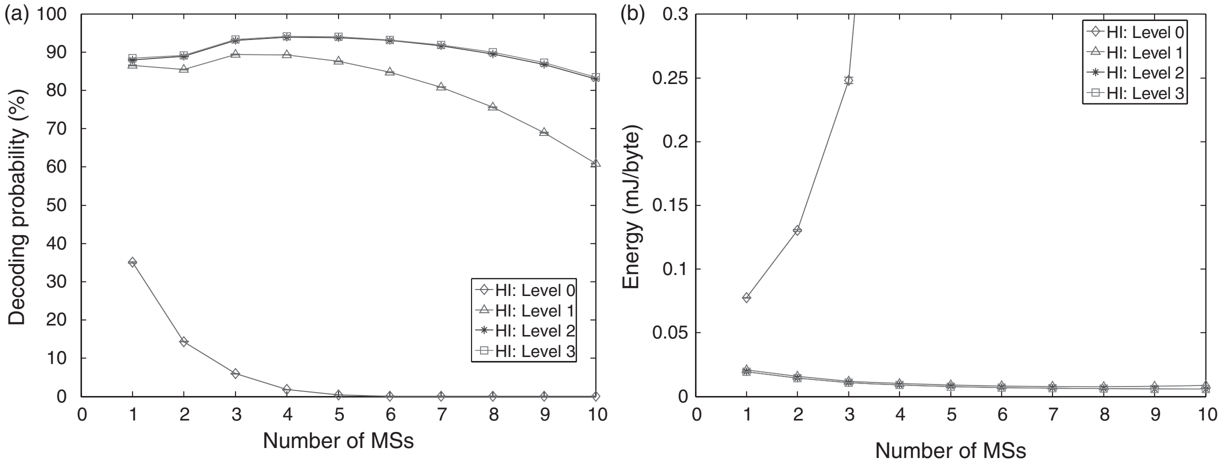

FIGURE 6. Decoding probability and energy efficiency vs. number of MSs, for different redundancy levels.

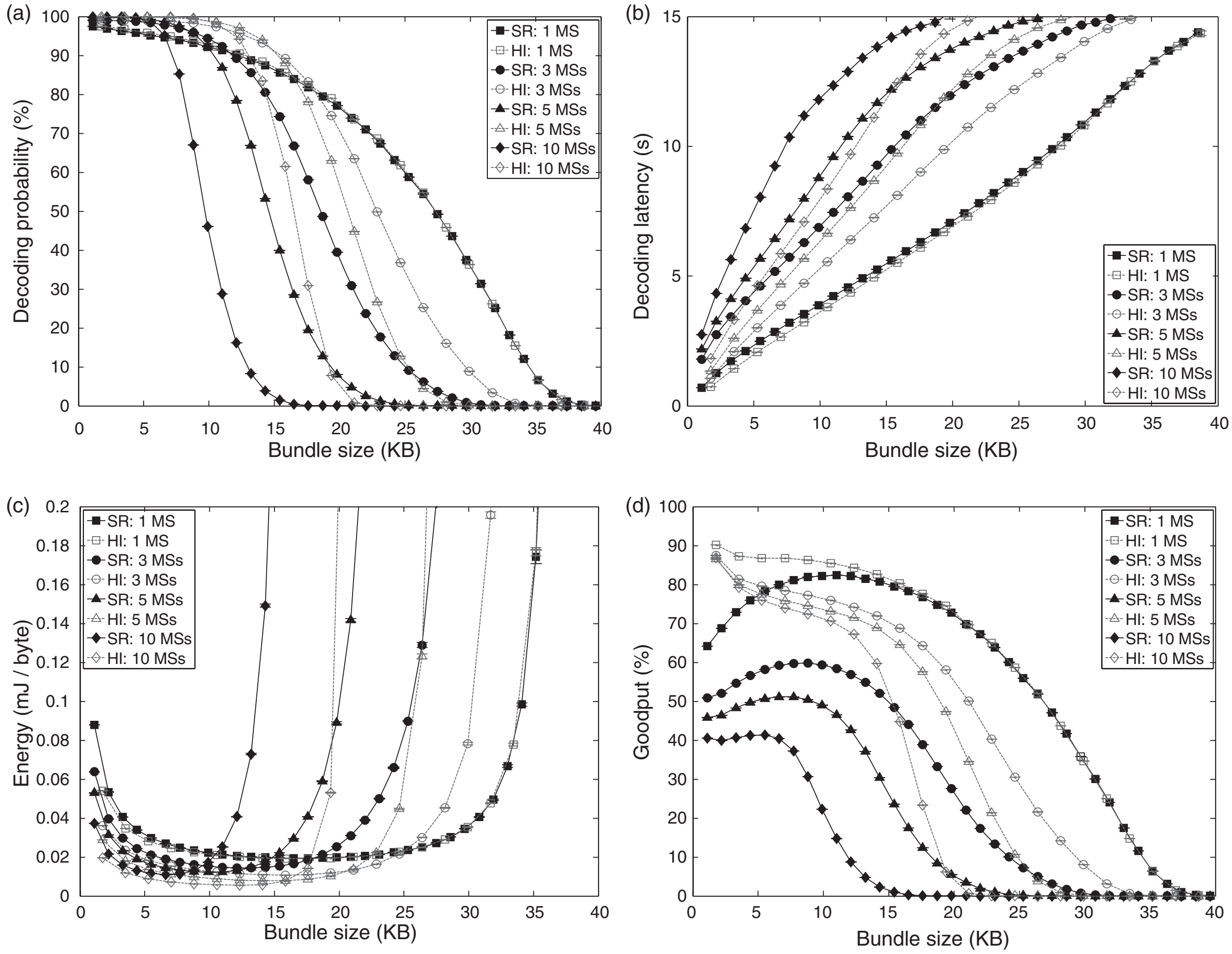

FIGURE 7. Performance comparison in the high mobility scenario.

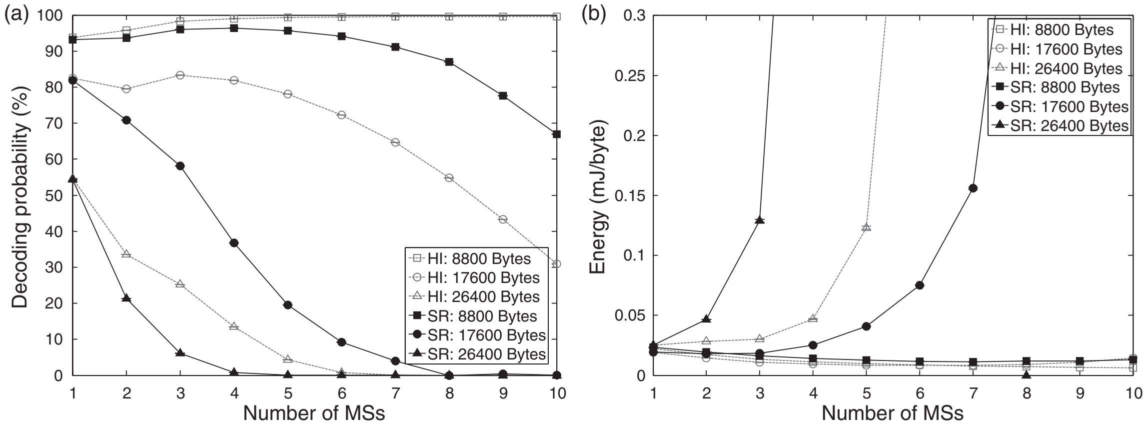

FIGURE 8. Decoding probability and energy efficiency vs. numbers of MSs.

by walking people) and, hence, their speed is assumed to be limited (3.6 km/h). Consequently, the contact time available for data delivery is very large (up to 158 s in our experiments).

Figure 9 summarizes the simulation results obtained in this scenario. In general, the trend is similar to the *High Mobility* scenario. When there is a single MS, the two protocols exhibit approximately the same performance. Instead, when the number

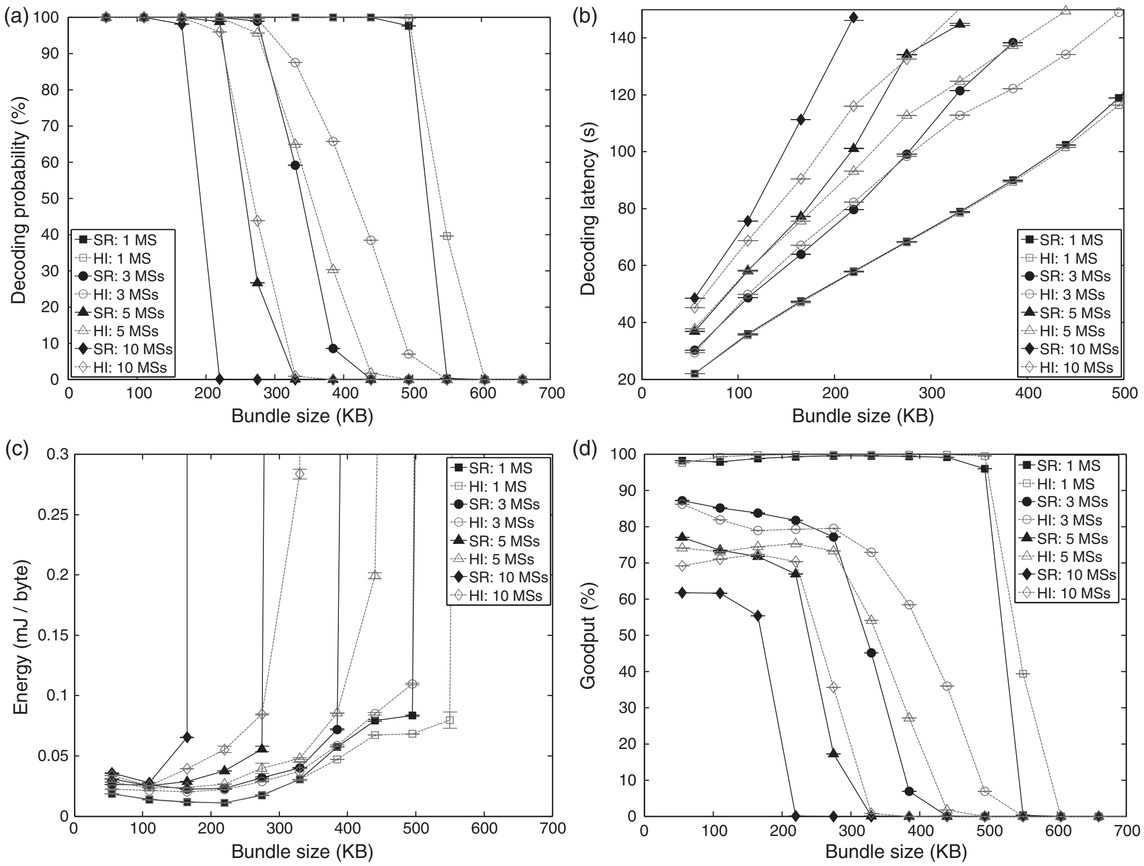

FIGURE 9. Performance comparison in the low mobility scenario.

of MSs is larger than one, HI outperforms SR with respect to all considered performance metrics, and the difference between corresponding curves increases with the number of MSs. Obviously, in the *Low Mobility* scenario the sensor node is able to transfer bundles of significantly larger size (up to 550 KB with a single MS) due to the larger contact time.

6.4. Heterogeneous mobility scenario

In this section, we complete our evaluation by analyzing the *Heterogeneous Mobility* scenario where MSs move at different speeds. This can be an urban scenario in which MSs are carried by different typologies of users (e.g. a pedestrian, a user moving in a car). Specifically, we consider three different speeds for MSs: high speed (i.e. 40 km/h), low speed (i.e. 3.6 km/h) and intermediate speed (i.e. 20 km/h).

This scenario is more complex than those presented in the previous sections. Having a scenario with different MS speeds corresponds to having different contact times. For example, there is about one order of magnitude between the contact time of MSs moving at the lowest speed and MSs moving at the highest speed (Table 1). In addition, an MS moving at the highest speed which enters last could be the first to leave the sensor area since it has the shortest contact time.

The presence of an MS moving at 40 km/h limits the size of the bundle to be transmitted. As we have shown in the *High Mobility* scenario (see Section 6.2), MSs are not able to correctly receive any information for bundle sizes larger than 40 KB, since their contact time is very limited. For this reason, in the simulation experiments presented in this section, we only consider bundle sizes smaller than 40 KB size. This guarantees that all MSs considered in the simulation are able to receive data distributed by the static sensor.

In our simulation experiment, three MSs enter the contact area at different times, as explained in Section 5.3. The three MSs move at different speeds (i.e. one at 40 kmh, one at 20 kmh, one at 3.6 kmh) and the sequence of MS speed in the experiment is chosen randomly.

As highlighted in Fig. 10, HI outperforms SR both in terms of decoding probability and consumed energy. Specifically, using HI increases the delivery probability by 22% on average and the energy saving by 40% on average wrt using SR. Focusing on the decoding probability (Fig. 10a), HI is able to manage higher

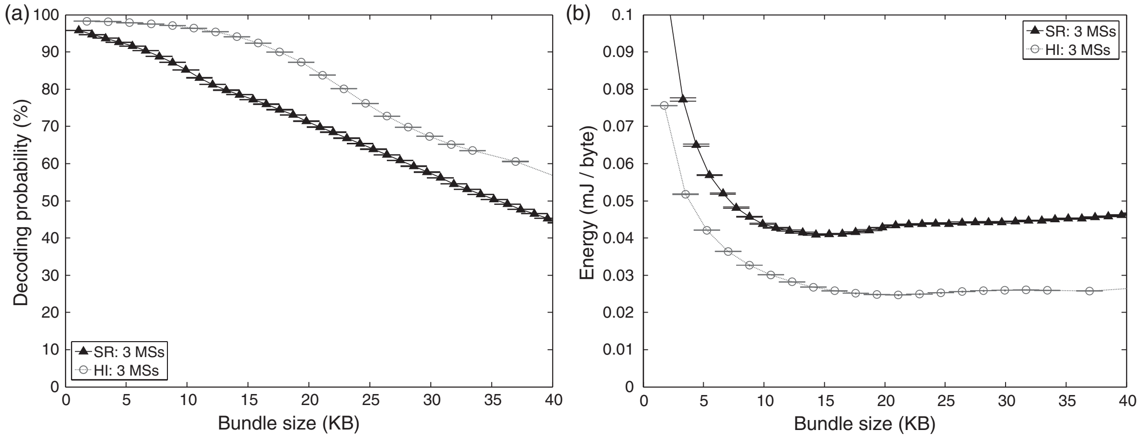

FIGURE 10. Performance comparison in the heterogeneous mobility scenario.

bundle sizes. For example, with reference to 90% decoding probability value, the HI bundle size is approximately 20 KB higher than the SR bundle size. In addition, unlike in the *High Mobility* scenario (see Section 6.2), here the decoding probability at 40 KB is ∼60% for HI and ∼45% for SR. This is due to the presence of the other two MSs that, moving at lower speeds, have a longer contact time and are able to receive higher bundle sizes. The results of Fig. 10 confirm that HI is very versatile and it is also suitable to heterogeneous environments characterized by groups of MSs moving at different speeds.

7. VALIDATION WITH REAL SENSOR NODES

As is well known, simulation experiments might not take into account all factors that can occur in a real environment due to the simplifying assumptions introduced in the simulation model. Hence, we decided to complement our analysis by means of a validation with real sensor nodes. To this end, we used the Tmote Sky sensor platform [34]. Tmote Sky sensor nodes use the Chipcon CC2420 radio transceiver which is compliant with the IEEE 802.15.4 physical layer [35] and enables a bit rate of 250 Kbps over the unlicensed 2.4 GHz ISM band.

TABLE 4. Tmote Sky sensor node's parameters [33].

Parameter	Value
Bit rate	250 kbps
Message/ACK size	110 bytes
Frame size	128 bytes
Transmission power at 0 dBm	52.2 mW
Reception power	56.4 mW
Idle power	3 μW

Table 4 summarizes the main operating parameters of Tmote Sky sensor nodes. All other parameter settings are as shown in Table 2. We want to emphasize that the value of 17 ms used in our simulation experiments for message and ACK transmission times (δ_{MSG} and δ_{ACK}, respectively) corresponds approximately to the average time required in Tmote Sky sensor nodes to transmit a 110-byte message[6].

In order to be able to compare real measurements with simulations, we must have the same packet loss model. Due the high variability of channel condition, it is almost impossible to obtain a real experiment with a packet loss comparable to that assumed in the simulations. Moreover, it is also important for the evaluation of the confidential intervals to generate i.i.d. experiments. This is not possible with real measurements. In addition, managing several MSs that simultaneously move at a predefined constant speed is not easy in practice. Therefore, we decided to adopt the approach described below. The sensor node acting as an MS is put at a short distance from the static sensor node (in the order of 1 m), without any obstacles in between. This allows a percentage of successful transmissions from the sensor node to the MS, and vice versa, of ∼100%. Then, to simulate the effect of message losses (and mobility as well) we used the same packet loss model considered in simulations. Received messages are discarded at the destination with a probability $p(t)$ given by expression (1). In the case of multiple MSs, we assumed that they travel along the same path but are separated by a random delay. Hence, they experience the same message loss probability function but with different timing as they are supposed to enter the contact area at different times.

The methodology and the performance metrics used during experiments are similar to those used in simulations (see Section 6) with some minor differences. Specifically, each

---

[6] To derive the average transmission time we transmitted a 110-byte message to a very close destination (1 m from the source node), for a very large number of times.

experiment has been repeated only five times, with each replica consisting of 120 contacts (in simulations we considered 10 000 contacts per replica), since generally performing real experiments is more complex and costly in effort and time than simulations. As in Section 6, the results presented below are averaged over all replicas. For the sake of space we only refer to the *High Mobility* scenario (i.e. MSs move at 40 km/h).

Figure 11a and b compares the decoding probabilities—derived through simulations and real measurements—of HI and SR, respectively. Similarly, Fig. 12a and b shows the energy efficiency of the two protocols. We performed experiments with a number of MSs varying in the range [1–5]. However, for clarity, in Figs 11 and 12 we only show results related to 1 and 5 MSs. For the sake of space we also omitted the comparison in terms of decoding latency and goodput. We can observe that simulation and experimental curves are generally very close to each other. Clearly, experimental results have a larger variability than simulation results, mainly due to the lower number of contacts considered in each experiment. However, the experimental results validate and confirm the simulation results presented in Section 6.

From the energy point of view, note that the energy consumption shown in Fig. 12 refers to the communication phase only. However, for a fair comparison among the two protocols, we have to consider also the encoding process, requested by HI, as it consumes energy at the sensor node. Note that in the simulation analysis presented in Section 6 this contribution has been neglected since it strictly depends on the technology used. On the contrary, this factor should be considered when performing experiments with a real testbed. To this aim, in the following we also investigated the impact of the encoding process in terms of energy on the Tmote Sky sensor platform. The energy consumed for decoding messages at the MS side is of less importance since MSs are not energy constrained and for this reason it is not included in the following discussion. We measured that 40.5 μJ/byte are needed (on average) when

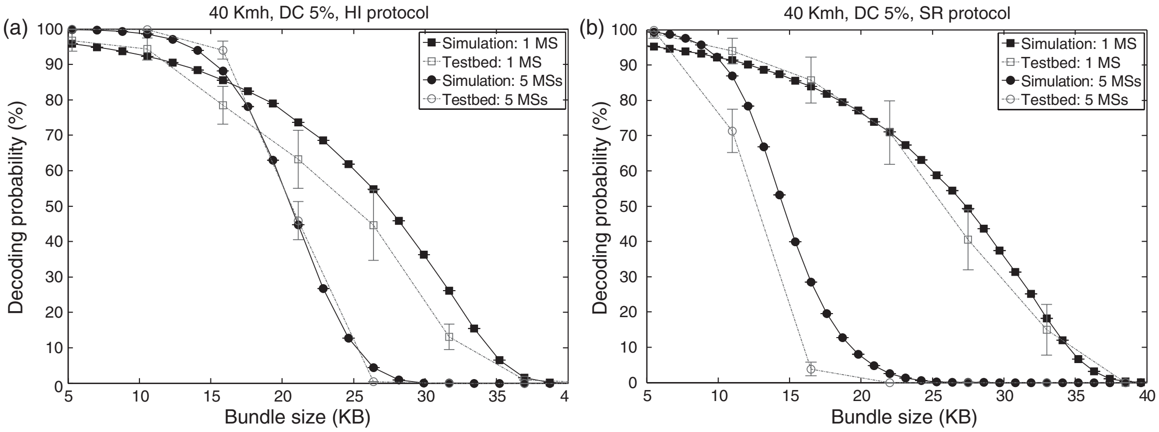

FIGURE 11. Decoding probability vs. bundle size for HI (a) and SR (b).

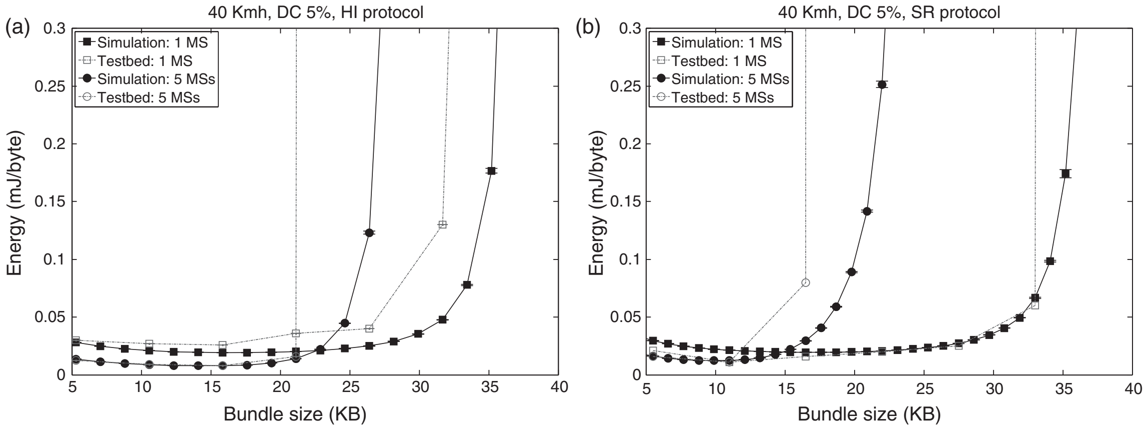

FIGURE 12. Energy vs. bundle size for HI (a) and SR (b).

using the highest level of redundancy (256 codes). In this case, the energy consumption due to the encoding phase cannot be ignored since it is of the same order of magnitude of the energy consumption shown in Fig. 12 (i.e. ∼30 μJ/byte for the one MS scenario). However, the energy cost of the encoding phase can be significantly reduced using a lower degree of redundancy. For example, only 3.9 μJ/byte are needed when using the redundancy Level 2 (i.e. 32 codes), which represents a negligible factor (i.e. one order lower) with respect to the energy required for the communication. Hence, in a Tmote Sky implementation, 32 codes are a good compromise between performance and energy consumption. This confirms the advantages of using the HI protocol in comparison with the SR protocol in sparse sensor networks.

8. DISCUSSION AND CONCLUSIONS

In this paper, we have investigated the problem of reliable and energy-efficient data delivery in sparse sensor networks with MSs. In particular, we have defined the HI data delivery protocol, a hybrid adaptive data-transfer protocol that combines efficiently Erasure Coding with ARQ. In HI, the encoding process is performed in advance by the sensor node so as to save useful resources (i.e. contact time). In addition, the protocol is able to adapt the number of codes to be transmitted based on message loss patterns experienced by MSs. Focusing on the transmission phase, we have compared the performance of the proposed data-transfer protocol with that of an alternative protocol based on a traditional ARQ scheme with Selective Retransmissions. In addition, we have also complemented our simulation analysis by means of an experimental validation performed with real sensor nodes using an IEEE 802.15.4-compliant physical layer. The obtained results have shown that the proposed data-transfer protocol largely outperforms the alternative protocol when there are multiple MSs. In addition, using the HI protocol is convenient also with a single MS, when the amount of data to be delivered is limited.

The protocol version considered in the above analysis is based on RS-codes and assumes to generate the maximum level of redundancy. Under such hypothesis, even in the maximum redundancy case, we have measured that the encoding process takes ∼26.5 ms to generate each code and, hence, a total time of ∼6.5 s to generate the 256 codes composing each block when using the Tmote Sky sensor platform. This time is negligible if compared with times characterizing the sparse network scenario. Since MSs interact sporadically with sensor nodes, the inter-contact times are in the order of (dozen of) minutes; hence the sensor node can produce the required redundancy much earlier than the next contact occurs and, as a consequence, not consume the limited contact time. For completeness note that the decoding process is not critical as the MS envisioned in such a scenario is typically resource rich. The most critical limitation imposed by the aforementioned platform is the available memory since it limits the size of the original bundle to be stored and the order of redundancy that can be added to the original data. In a real implementation this problem can be easily overcome taking into account a lower level of redundancy. In Section 6.1, we have shown that 32 codes (i.e. Level 2) guarantee a near optimal performance and are affordable with the standard resource of the sensors currently available. Furthermore, since the technology is continuously evolving, sensor memory will increase further. For example, more recent sensor platforms have increased the memory capabilities at least by one order (e.g. 96 KB and 512 KB for the Jennic[7] and Sun Spot[8] sensor platforms, respectively), hence guaranteeing the feasibility of using RS-codes in real sensor nodes, also in the case of larger bundle size.

Finally, note that any other encoding scheme that runs efficiently in sensor nodes with limited computational and memory capabilities can be accommodated in our proposed protocol with minor modifications.

ACNOWLEDGEMENTS

The authors wish to express their gratitude to Luciana Pelusi, Monica Castronuovo and Mario Di Francesco for their useful comments and suggestions.

FUNDING

This work has been supported partially by the European Commission under the BIONETS (027748) FET FP6 project, and partially by the Italian Ministry for Education and Scientific Research (MIUR) under the PRIN WiSe DeMon projects.

REFERENCES

[1] Akyildiz, I., Su, W., Sankarasubramaniam, Y. and Cayirci, E. (2002) A survey on wireless sensor networks: A survey. *Comput. Netw. (Elsevier) J.*, **38**, 393–422.

[2] Reed, I.S. and Solomon, G. (1960) Polynomial codes over certain finite fields. *J. Soc. Ind. Appl. Math.*, **8**, 300–304.

[3] Ekici, E., Gu, Y. and Bozdag, D. (2006) Mobility-based communication in wireless sensor networks. *IEEE Commun. Mag.*, **44**, 56–62.

[4] Anastasi, G., Conti, M., Passarella, A. and Pelusi, L. (2008) Mobile-Relay Forwarding in Opportunistic Networks. In Ibnkahla, M. (ed.), *Adaptation and Cross Layer Design in Wireless Networks.* CRC Press, New York, NY.

[5] Chakrabarti, A., Sabharwal, A. and Aazhang, B. (2003) Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks. *Proc. 2nd Int. Workshop on Information Processing in Sensor Networks (IPSN 2003).* Palo Alto, CA, April 22–23, pp. 129–145. Springer Berlin, Heidelberg.

---

[7] http://www.jennic.com/.

[8] http://www.sunspotworld.com/.

[6] Shah, R., Roy, S., Jain, S. and Brunette, W. (2003) Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks. *Proc. 1st IEEE Int. Workshop on Sensor Network Protocol and Applications (SNPA 2003)*, Anchorage, AK, May 11, pp. 30–41. IEEE Computer Society Press.

[7] Somasundara, A., Kansal, A., Jea, D., Estrin, D. and Srivastava, M. (2006) Controllably mobile infrastructure for low energy embedded networks. *IEEE Trans. Mob. Comput.*, **5**, 958–973.

[8] Zhao, W and Ammar, M. (2003) Message Ferrying: Proactive Routing in Higly-partitioned Wireless Ad Hoc Networks. *Proc. IEEE Workshop on Future Trends in Distributed Computing Systems (FTDCS 2003)*, San Juan, Puerto Rico, May 28–30, pp 308–314. IEEE Computer Society Press.

[9] Jain, S., Shah, R.C., Brunette, W., Borriello, G. and Roy, S. (2006) Exploiting mobility for energy efficient data collection in wireless sensor networks. *Mob. Netw. Appl.*, **11**, 327–339.

[10] Jea, D., Somasundra, A. and Srivastava, M. (2005) Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks. *Proc. IEEE/ACM Int. Conf. Distributed Computing in Sensor Systems (DCOSS 2005)*, Marina del Rey, CA, June 30–July 1, pp. 244–257. Springer.

[11] Kansal, A., Somasundara, A., Jea, D. and Srivastava, M.B. (2004) Intelligent Fluid Infrastructure for Embedded Networks. *Proc. 2nd ACM Int. Conf. Mobile Systems, Applications and Services (MobiSys 2004)*, Boston, MA, June 6–9, pp. 99–110. ACM Press.

[12] Song, L. and Hatzinakos, D. (2005) Dense Wireless Sensor Networks with Mobile Sinks. *Proc. IEEE Conf. Acoustic, Speech, and Signal Processing (ICASSP 2005)*, Philadelphia, PA, March 18–23, pp. 677–680. IEEE Computer Society Press.

[13] Song, L. and Hatzinakos, D. (2007) Architecture of wireless sensor networks with mobile sinks: Sparsely deployed sensors. *IEEE Trans. Veh. Technol.*, **56**, 1826–1836.

[14] Anastasi, G., Conti, M., Monaldi, E. and Passarella, A. (2007) An Adaptive Data-Transfer Protocol for Sensor Networks with Data Mules. *Proc. IEEE Int. Symp. a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007)*, Helsinki, Finland, June 18–21, pp. 1–8. IEEE Computer Society Press.

[15] Basagni, S., Carosi, A., Melachrinoudis, E., Petrioli, C. and Wang, M. (2008) Controlled sink mobility for prolonging wireless sensor networks lifetime. *ACM Wirel. Netw. J. (WINET)*, **14**, pp. 831–858, Springer US.

[16] Shi, G., Liao, M., Ma, M. and Shu, Y. (2008) Exploiting Sink Movement for Energy-Efficient Load-Balancing in Wireless Sensor Networks. *Proc. 1st ACM Int. Workshop on Foundations of Wireless Ad hoc and Sensor Networking and Computing (FOWANC 2008)*, Hong Kong, China, May 28, pp. 39–44. ACM Press, New York, NY.

[17] Pelusi, L., Passarella, A. and Conti, M. (2008) Encoding for Efficient Data Distribution in Multihop Ad Hoc Networks. In Boukerche, A. (ed.), *Algorithms and Protocols for Wireless, Mobile Ad Hoc Networks*. Wiley-IEEE Press.

[18] Kim, S., Fonseca, R. and Culler, D. (2004) Reliable Transfer in Wireless Sensor Networks. *Proc. IEEE Int. Conf. Sensor and Ad hoc Communications and Networks (SECON 2004)*, S. Clara, CA, October 4–7, pp 449–459. IEEE Computer Society Press.

[19] Dimakis, A.G., Prabhakaran, V. and Ramchandran, K. (2005) Ubiquitous Access to Distributed Data in Large-Scale Sensor Networks through Decentralized Erasure Codes. *Proc. 4th Int. Conf. Information Processing in Sensor Networks (IPSN 2005)*, Los Angeles, CA, April 15, pp. 111–117. IEEE Computer Society Press.

[20] Karlsson, P., Öberg, L. and Xu, Y. (2005) An Address Coding Scheme for Wireless Sensor Networks. *Proc. Scandinavian Workshop on Wireless Ad-hoc Networks (ADHOC '05)*, Stockholm, Sweden, May 3–4. IEEE Computer Society Press.

[21] Wen, H., Lin, C., Ren, F., Yue, Y. and Huang, X. (2007) Retransmission or Redundancy: Transmission Reliability in Wireless Sensor Networks. *Proc. IEEE Int. Conf. Mobile Ad Hoc and Sensor Systems (MASS 2007)*, Pisa, Italy, October 8–11, pp. 1–7. IEEE Computer Society Press.

[22] Xie, P. and Cui, J. An FEC-based Reliable Data Transport Protocol for Underwater Sensor Networks. *Proc. IEEE Int. Conf. Computer Communications and Networks (ICCCN 2007)*, Honolulu, Hawaii, August 13–16, pp. 747–753. IEEE Communication Society Press.

[23] Deb, S., Effros, M., Ho, T., Karger, D.R., Koetter, R., Lun, D.S., Médard, M. and Ratnakar, N. (2005) Network Coding for Wireless Applications: A Brief Tutorial. *Proc.. Int. Workshop on Wireless Ad-hoc Networks (IWWAN 2005)*, London, UK, May 23–26. Curran Associates, Inc.

[24] Widmer, J. and Le Boudec, J.-Y. (2005) Network Coding for Efficient Communication in Extreme Networks. *Proc. ACM SIGCOMM Workshop on Delay Tolerant Networking 2005*, Philadelphia, PA, August 22–26, pp. 284–291. ACM New York Press, NY.

[25] Vellambi, B., Subramanian, R., Fekri, F. and Ammar, M. (2007) Reliable and Efficient Message Delivery in Delay Tolerant Networks using Rateless Codes. *Proc. ACM/SIGMOBILE Int. MobiSys Workshop on Mobile Opportunistic Networking (MobiOpp 2007)*, S. Juan, Puerto Rico, June 11, pp. 91–98. ACM Press.

[26] Rizzo, L. and Vicisano, L. (1998) RMDP: An FEC-based reliable multicast protocol for wireless environments. *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, **2**, 23–31.

[27] Cerf, V. *et al.* (2007) *Delay-tolerant network architecture*. IETF RFC4838. http://tools.ietf.org/html/draft-irtf-dtnrg-arch-08.

[28] Byers, J.B., Luby, M. and Mitzenmacher, M. (2002) A digital fountain approach to asynchronous reliable multicast. *IEEE J. Sel. Areas Commun.*, **20**, 1528–1540.

[29] Rizzo, L. (1997) Effective erasure codes for reliable computer communication protocols. *ACM Comput. Commun. Rev.*, **27**, 24–36.

[30] Luby, M. (2002) LT Codes. *Proc. 43rd Annual IEEE Symp. Foundations of Computer Science (FOCS2002)*, Vancouver, Canada, November 16–19, pp. 271–280. IEEE Computer Society Press.

[31] Maymounkov, M. (2002) Online Codes. Technical Report TR2002-833. http://cs.nyu.edu/web/Research/TechReports/TR2002-833/TR2002-833.pdf.

[32] Kurose, J.F. and Ross, K.W. (2008) *Computer Networking: A Top-Down Approach, 4/E*. Addison-Wesley.

[33] Anastasi, G., Conti, M., Gregori, E., Spagoni, C. and Valente, G. (2007) Motes sensor networks in dynamic scenarios: An experimental study for pervasive applications in urban environments. *Int. J. Ubiquitous Comput. Intell.*, **1**, 9–16.

[34] Tmote Sky Platform, *MoteIV corporation.* http://www.moteiv.com/products/tmotesky.php.

[35] LAN/MAN Standards Committee of the IEEE Compter Society (2003) IEEE 802.15.4, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). IEEE Std 802.15.4™-2003. http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf.

APPENDIX: RS-CODES

RS-codes [2] are a form of (n, k)-codes. Assume that a source data message is a *word* and let a sequence of k words be represented by a vector, say $\mathbf{x}$, of k elements. Encoding is represented by an encoding function $f(\cdot)$ which is applied to $\mathbf{x}$ and produces an encoded vector of n codewords. When the encoding function is *linear*, the code is said to be linear too. In the following a brief introduction to general linear codes will be given and then the focus will be on RS-codes as they represent a special case of linear codes.

A.1. Linear codes

In *linear codes* the encoding function is linear and can be represented by a matrix $\mathbf{G}$, throughout referred to as *encoding matrix*. Hence, encoding corresponds to working out a matrix-by-vector multiplication. Given a vector $\mathbf{x}$ of original words produced by the source node, the corresponding vector of codewords, $\mathbf{y}$, is obtained as follows:

$$\mathbf{y} = \mathbf{G} \cdot \mathbf{x} \tag{A.1}$$

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} g_{0,0} & g_{0,1} & \cdots & g_{0,k-1} \\ g_{1,0} & g_{1,1} & \cdots & g_{1,k-1} \\ g_{2,0} & g_{2,1} & \cdots & g_{2,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n-1,0} & g_{n-1,1} & \cdots & g_{n-1,k-1} \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{k-1} \end{pmatrix} \tag{A.2}$$

where $\mathbf{x} = (x_0 \quad x_1 \quad \cdots \quad x_{k-1})^T$ is the vector of k source words, $\mathbf{y} = (y_0 \quad y_1 \quad \cdots \quad y_{n-1})^T$ the vector of n codewords and $\mathbf{G}_{(n\times k)}$ the encoding matrix. The destination node can decode the original data once it has received k out of the n codewords totally produced. Let $\mathbf{y}'$ be the vector of the k codewords received, and $\mathbf{G}'$ its encoding sub-matrix. Then

$$\mathbf{y}' = \mathbf{G}' \cdot \mathbf{x} \tag{A.3}$$

$$\begin{pmatrix} y_{i,0} \\ y_{j,1} \\ \vdots \\ y_{l,n-1} \end{pmatrix} = \begin{pmatrix} g_{0,0} & g_{0,1} & \cdots & g_{0,k-1} \\ g_{1,0} & g_{1,1} & \cdots & g_{1,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n-1,0} & g_{n-1,1} & \cdots & g_{n-1,k-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \end{pmatrix} \tag{A.4}$$

The encoding sub-matrix $\mathbf{G}'_{(n\times k)}$ is a $k \times k$ matrix obtained by extracting from the encoding matrix $\mathbf{G}_{(n\times k)}$ those rows that correspond to the elements of vector $\mathbf{y}'$. Thus, for example, if the j-th codeword (i.e. y_j) of original vector of codewords is inserted as the second element in vector $\mathbf{y}'$ (i.e. $y_{j,1}$), then the j-th row of matrix $\mathbf{G}_{(n\times k)}$ is picked up and inserted as the second row in matrix $\mathbf{G}'$. Clearly, decoding means finding out the solution of the linear equation $\mathbf{y}' = \mathbf{G}' \cdot \mathbf{x}$, as follows.

$$\mathbf{x} = \mathbf{G}'^{-1} \cdot \mathbf{y} \tag{A.5}$$

Note that the destination must be sure to identify the rows in $\mathbf{G}_{(n\times k)}$ corresponding to any received element of $\mathbf{y}$, and that the set of rows corresponding to $\mathbf{y}'$ must be linearly independent. As is clear, for the decoding to be possible, the encoding matrix $\mathbf{G}$ must have rank k.

A.2. Encoding process of RS-codes

RS-codes are a subset of linear codes. Source words are seen as the coefficients of a polynomial of degree $k - 1$, whereas codewords are seen as values of the polynomial worked out at n different points that can be chosen arbitrarily. Let the polynomial be as follows:

$$p(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{k-1}x^{k-1} \tag{A.6}$$

where $a_0, a_1, \ldots, a_{k-1}$ are the k words generated at the source for transmission and $p(x)$ is a single codeword obtained by evaluating the polynomial at point $\mathbf{x}$. The encoding process for an RS (n, k)-code is thus as follows:

$$\begin{pmatrix} p(x_0) \\ p(x_1) \\ p(x_2) \\ \vdots \\ p(x_{n-1}) \end{pmatrix} = \begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{k-1} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{k-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{k-1} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{k-1} \end{pmatrix} \tag{A.7}$$

where $x_0, x_1, \ldots, x_{n-1}$ are the n points selected for evaluation of the polynomial. They can be chosen arbitrarily, for example, for simplicity of encoding, or alternatively they can be all possible integer values that can be represented over the number of bits available. The encoding matrix of RS-codes is characterized by a geometric progression in each row. Such matrices are named *Vandermonde* matrices. When codewords include a verbatim copy of the source words, the code is said to be *systematic*. This corresponds to including the identity matrix $\mathbf{I}_k$ in the encoding matrix. The advantage of using a systematic code is that it simplifies the reconstruction of source words in case very few losses are expected. If, for example, only two (out of k) received codewords are original words, the system of equations that must be solved to reconstruct the original words includes $k-2$ equations instead of k.

A.3. Decoding process of RS-codes

The decoding process of RS-codes consists in reconstructing all polynomial coefficients $a_0, a_1, \ldots, a_{k-1}$ in a unique way. Hence, the receiver has to receive k codewords which provide the polynomial value at exactly k points. Assuming that the identity (e.g. the sequence number) of codewords already received at the destination is known, the coefficients of polynomial can be computed by solving the following system:

$$\begin{pmatrix} y_{i,0} \\ y_{j,1} \\ \vdots \\ y_{l,n-1} \end{pmatrix} = \begin{pmatrix} 1 & x_{i,0} & x_{i,0}^2 & \cdots & x_{i,0}^{k-1} \\ 1 & x_{j,1} & x_{j,1}^2 & \cdots & x_{j,1}^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{l,k-1} & x_{l,k-1}^2 & \cdots & x_{l,k-1}^{k-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} \tag{A.8}$$

The decoding process matrix is the sub-matrix of the encoding matrix obtained by selecting the k rows which correspond to the codewords arrived (the i-th, j-th and the l-th rows in the example). The system admits a solution if the matrix is non-singular. The determinant of a $k \times k$ Vandermonde matrix has the following expression:

$$det(V) = \prod_{0 \leqslant l \leqslant t \leqslant k} (\hat{x}_t - \hat{x}_l) \tag{A.9}$$

$\hat{\mathbf{x}} = (\hat{x}_0 \quad \hat{x}_1 \quad \ldots \quad \hat{x}_{k-1})^{\mathrm{T}} = (x_{i,0} \quad x_{j,1} \quad \ldots \quad x_{l,k-1})^{\mathrm{T}}$ is the second column of the Vandermonde matrix. Hence, the determinant is non-null if and only if all the $\hat{x}_i$ ($i = 0, 1, \ldots, k-1$) are non-null and different from each other. Finally note that, to allow decoding RS-codes, both the source and destination nodes must know the encoding matrix.