Data Collection in Sensor Networks with Data Mules: an Integrated Simulation Analysis

Giuseppe Anastasi[*], Marco Conti[#], Mario Di Francesco[*]
Pervasive Computing & Networking Lab. (PerLab)

[*]*Dept. of Information Engineering*
University of Pisa, Italy
{firstname.lastname}@iet.unipi.it

[#]*CNR-IIT*
National Research Council, Italy
{firstname.lastname}@iit.cnr.it

Abstract

Wireless sensor networks (WSNs) have emerged as the enabling technology for a wide range of applications. In the context of environmental monitoring, especially in urban scenarios, a mobile data collector (data mule) can be exploited to get data sensed by a number of nodes sparsely deployed in the sensing field. In this paper we describe and analyze protocols for reliable and energy-efficient data collection in WSNs with data mules. Our main contribution is the joint performance analysis of the discovery and the data transfer phases of the data collection process. Our results show that a low duty cycle (i.e. in the order of 1%) is actually feasible for most common environmental monitoring applications. We also found that, depending on the mobility pattern of the data mule, a lower duty cycle may not be always a more convenient option for energy efficiency. Based on these results, we outline in the paper possible directions for improving the energy efficiency of data collection in sparse WSNs with data mules.

1. Introduction

The set of potential applications of wireless sensor networks is extremely large. However, environmental monitoring represents a class of applications that can particularly benefit from sensor networks [13]. In such applications a large number of sensor nodes is typically deployed over a geographical area to form a dense ad hoc network. Sensors use multi-hop communication to send data acquired from the external environment to an Access Point (AP) in the infrastructure (or a sink node). However, many environmental monitoring applications, such as monitoring of weather conditions in large parks, air quality in urban areas, terrain conditions for precision agriculture, and so on, do not require a fine-grain sensing and, thus, a sparse sensor network would be enough. This reduces costs since a lower number of devices is needed. However, as the distance between neighboring nodes becomes larger and larger, the communication is no longer possible, or requires too much energy. In other scenarios, the monitored area can be far away from the nearest AP, and deploying additional sensors for relaying data becomes too costly.

Data collection in such sensor networks can be achieved more efficiently by using *data mules* (or mules for short), i.e., mobile relay nodes that carry data from static sensor nodes to an infra-structured AP [8] (see Figure 1). Depending on the application scenario, mules may be either part of the external environment [4], [14] (e.g., buses, cabs, or walking people), or part of the network infrastructure [9], [11] (e.g., mobile robots). They visit sensor nodes at predictable or random times (depending on their nature and mobility pattern), pick up data, and carry them to an AP.

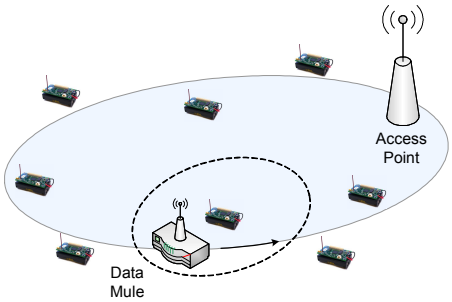


Figure 1. Data mule architecture

Mules are assumed to be power renewable, while sensor nodes are typically energy-constrained. Therefore, both the mule discovery process (by which a sensor detects that a mule is within its communication range), and the data transfer process (by which the

Work funded partially by the IST program of the European Commission under the FET-SAC HAGGLE project, and partially by the Italian Ministry for Education and Scientific Research (MIUR) under the FIRB ArtDeco project.

sensor transfers its data to the mule) must be energy efficient to prolong the lifetime of the sensor nodes. As the radio component is usually the major source of energy consumption, the total time during which the radio must be on should be minimized. On the other hand, if sensors are sleeping for most of the time, they could miss the passage of the mule so that they cannot transfer data until the next contact. In this case, the energy conservation gain may be pointless, because the sensor may transfer only a little amount of data so that the application reliability/quality requirements are not met.

Almost all solutions proposed in the literature for sparse WSNs with data mules actually have focused on the mule itself (i.e. [6], [7] and [8]), while a little attention has been devoted to the performance of the overall data collection process, especially in terms of energy consumption. Generally, simple schemes are used for both discovery (e.g., the periodic wakeup scheme of [11] and [14]) and data transfer (e.g., the stop-and-wait protocol used in [11] and [15]). In addition, often simplistic assumptions are made for both the message loss model (in opposition with real measurements such as [1]) and the mule mobility pattern (e.g., the strict schedule assumed in [4]). More recently, an ARQ window-based scheme has been introduced in [2], where it is analytically shown that using a window larger than one may significantly improve the performance of the data transfer protocol, thus reducing the energy consumption of the sensor node. However, [2] does not consider the impact of the mule discovery on the subsequent data transfer phase at all. A different approach [3] has been proposed in terms of transmission scheduling, i.e. when a node should wakeup and transmit to the mule. However, the solution proposed in [3] is evaluated only under the assumption that the mule follows a random-waypoint mobility pattern.

In this paper we consider the joint impact of discovery and data transfer protocols for energy-efficient data collection in sparse WSNs with data mules. We use a realistic message loss model derived from real measurements [1] and provide an integrated performance evaluation. In addition, our analysis is general as we do not assume a specific mobility pattern of the mule, although we discuss the impact on the data collection process of different parameters related to the mule mobility. The rest of the paper is organized as follows. Section 2 describes energy-efficient protocols for discovery and data transfer. Section 3 outlines the simulation setup and introduces relevant metrics for evaluation. Section 4 discusses the obtained results. Finally, Section 5 concludes the paper.

2. Discovery and Data Transfer Protocols Description

In this section we briefly describe the behavior of the sensor nodes and the data mule and, specifically, the discovery and the communication protocols they use for energy-efficient data collection. In the following, we only assume the network is sparse enough so that at any time at most only a single static sensor node is in the contact area covered by the mule.

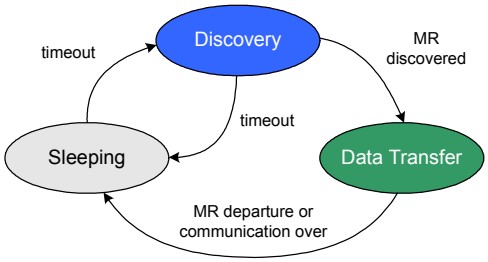


Figure 2. State diagram for the sensor node

The overall data collection process can be split into three main phases. Figure 2 shows the state diagram of the static sensor node [10]. As mule arrivals are generally unpredictable, the static node initially performs a discovery phase for the timely detection of the mule. As soon as the mule presence has been detected, the static node moves from the discovery state to the data transfer state, in which sensed data are transferred to the mule itself. When the data transfer phase is over, the static node may switch to the discovery state again in order to detect the next mule passage. However, if the mule has a (even partially) predictable mobility, the static node can exploit this knowledge to further reduce its energy consumption [10]. In this case, the static node goes to a sleep state until the next expected arrival of the mule within the contact area. Otherwise, when no knowledge on the mule mobility pattern is available, the static node immediately enters the discovery state.

We now briefly describe the discovery and data transfer protocols used by the static sensor and the data mule in the corresponding phases. To allow a timely discovery, the mule periodically advertises its presence by sending special messages called *beacons*. The duration of a beacon is T_D, while the time between the transmission of subsequent beacons (*beacon period*) is T_B. As mule arrivals may be separated by a long time, while in the discovery state the static node operates with a low duty cycle δ defined by the activity and sleep times T_{ON} and T_{OFF}, respectively, in order to save energy. The static node follows a periodic wakeup

scheme, with its activity time set to $T_{ON} = T_D + T_B$. The rationale behind this choice is to ensure the node to actually receive a complete beacon during its activity time, provided that it wakes up when the mule is in the contact area. The T_{OFF} parameter, instead, is set to the value which enforces the desired duty cycle $\delta = T_{ON} / (T_{ON} + T_{OFF})$.

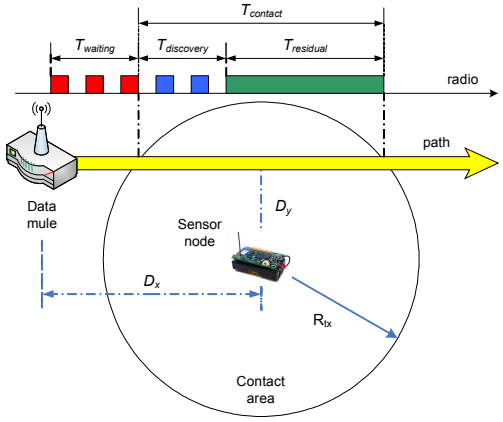


Figure 3. Reference scenario

As soon as the static node receives a beacon from the mule, it switches to the always-on mode – i.e., to a 100% duty cycle – and enters the data transfer state. The static node uses the full duty cycle to reduce the period devoted to data transfer and also the probability of losing contact with the mule while transferring data. On the other hand, the mule enters the data transfer phase immediately after receiving the first message sent by the static node. During data transfer an ARQ (Automatic Repeat reQuest) based communication scheme with selective retransmission is adopted. In detail, the static node sends W consecutive messages where the duration of each message (the slot size) is the same, and W (the window size) is assumed to be fixed and known at both the sender and the receiver. After sending all messages in the window, the static node waits for the corresponding acknowledgement, which includes a mask for notifying correctly received messages within the last window. Then the static node sends the next W messages including both missed messages as well as new ones. As the acknowledgement may get lost as well, the static node retransmits all the messages of the previous window in the case it does not receive the acknowledgement within a slot. The data transfer phase ends either when all buffered data has been sent by the static node, or the mule has exited the contact area. Also the mule ends its data transfer phase when it does not receive any more message in a given period of time.

3. Simulation Setup

In our simulation analysis we refer to the simple network scenario shown in Figure 3. Since we assume the network is sparse, we will consider a single static sensor node and a single mule. We also assume that the mule moves along a linear path at a fixed vertical distance (D_y) from the node, at a constant speed v. In addition, we assume that the data transfer phase spans over the entire residual contact time.

In our analysis we used a message loss function derived from the experimental data presented in [1] and measured in the same scenario described above. To get a more flexible model, we derived a polynomial interpolation of measured data in the form

$$p(D_x) = a_2 \cdot D_x^2 + a_1 \cdot D_x + a_0 \qquad (1)$$

Specifically, $p(D_x)$ provides the probability that a message transmitted when the data mule is at a horizontal distance D_x (see Figure 3) is lost. To derive the coefficients – reported in Table 1 for different mule speeds v and for a vertical distance $D_y = 15\text{m}$ – we used the same methodology described in [2]. We used the message loss function (1) only within the contact area, defined as the region in which the mule and the static node are in the communication range of each other.

Table 1. Message loss polynomial coefficients for different mule speeds (D_y=15 m)

Coefficient	v=3.6 Km/h	v=20 Km/h	v=40 Km/h
a_0	0.133	0.3828	0.4492
a_1 (m^{-1})	0	0	0
a_2 (m^{-2})	$0.138 \cdot 10^{-3}$	$9.072 \cdot 10^{-4}$	$6.237 \cdot 10^{-5}$

Based on this realistic message loss model, we performed an integrated performance evaluation of the discovery and the data transfer phases. Before discussing the performance metrics considered in our analysis, it may be worthwhile to introduce the following definitions (see Figure 3).

- *Contact Time* ($T_{contact}$), denotes the time taken by the mule to traverse the communication range of the static node.
- *Waiting Time* ($T_{waiting}$), denotes the time interval between the instant when the static node enters the discovery state and the instant when the mule enters the communication range of the static node.
- *Discovery Time* ($T_{discovery}$), defined as the time from when the mule enters the communication range of the static node to when it is actually detected by the static node itself.

- *Residual Contact Time* ($T_{residual}$), denotes the time from the instant when the mule is discovered to the instant when it exits the communication range of the static node. Obviously, $T_{residual} = T_{contact} - T_{discovery}$.

Clearly, all the above quantities are random variables.

We can now introduce the performance metrics, which are defined as follows.

- *Residual Contact Ratio*, defined as the ratio between the average Residual Contact Time and the average Contact Time.
- *Contact Miss Ratio*, defined as the fraction of mule passages not detected by the static sensor (i.e. $T_{residual} = 0$).
- *Throughput*, defined as the average number of bytes/messages correctly transferred to the mule at each passage of the mule itself.
- *Energy Consumption per Byte*, defined as the mean energy spent by the static sensor per each byte correctly transferred to the mule.

The Energy Consumption per Byte is calculated as $E_{byte} = \frac{P_{rx} \cdot T_{rx} + P_{tx} \cdot T_{tx} + P_{sleep} \cdot T_{sleep}}{n_{bytes}}$, where P_{state} and T_{state} denote the power drained and the time spent in a given radio state (i.e. receive, transmit and sleep), respectively. It is $T_{sleep} = (T_{waiting} + T_{discovery}) \cdot (1 - \delta)$, $T_{tx} = T_{residual} \cdot \left(\frac{W}{W+1}\right)$, and $T_{rx} = (T_{waiting} + T_{discovery}) \cdot \delta + T_{residual} \cdot \left(\frac{1}{W+1}\right)$. These expressions can be justified as follows. While in the discovery state $T_{waiting} + T_{discovery}$ the static node is active (for receiving possible beacons) for a fraction given by the duty cycle δ. While in the data transfer state $T_{residual}$ the static node is always active and, cyclically, transmits W consecutive messages and waits for the related acknowledgement.

4. Simulation Results

We implemented the discovery and the data transfer protocols in the TOSSIM simulator [12] and ran a set of experiments considering different values for the mule speed and the duty cycle of the static node. We introduced the message loss function (1) in our TOSSIM simulation model. We considered 3.6 Km/h, 20 Km/h and 40 Km/h as representative speeds of mules in a typical urban scenario (e.g. pedestrians, shuttles and buses). For the sake of space, we will only present results related to 3.6 and 40 Km/h below. In the following, we assume the static node is equipped with a Chipcon CC1000 radio (which is used in MICA2 series motes [5]). The average contact time is 157s, and 18s at 3.6 and 40 Km/h, respectively. All other simulation parameters are summarized in Table 2.

To increase the accuracy of the simulation results, we used the replication method. We replicated each simulation run (consisting of 50 mule passages) 5 times so as to simulate 250 passages per each experiment. We derived confidence intervals with a 90% confidence level.

Table 2. Simulation parameters

Parameter	Value
Sleep power	0.6 μW
Transmit power (0 dBm)	49.5 mW
Receive (idle) power	28.8 mW
Bit rate	19.2 Kbps
Message payload size	24 bytes
Frame size	36 bytes
Slot size	15 ms

4.1 Discovery phase

In the first set of experiments we evaluated the effectiveness of the discovery protocol in terms of both Residual Contact Ratio and Missed Contact Ratio. Since the timely mule discovery strongly depends on the beaconing rate, we varied the beacon period T_B in the set 0, 100, 300 and 500 ms. The special case of $T_B = 0$ (throughout referred to as *back-to-back*) represents the case in which beacons are sent one just after the other. The back-to-back beaconing scheme is optimal for discovery, because it allows the static node to detect the mule passage as soon as possible. Unfortunately, this scheme is not practical, because the static sensor might not be able to reply to the beacon message as the channel is always busy. However, the back-to-back scheme is useful for comparison.

Figure 4-a and Figure 4-b show the Residual Contact Ratios for different beacon periods and duty cycles when the mule moves at 3.6 Km/h and 40 Km/h, respectively. We can see the static node rapidly discovers the mule when it moves slowly (i.e. at 3.6 Km/h), regardless of its duty cycle. In detail, we can see that the difference between the back-to-back and the other beaconing rates is not high for the 10% and the 1% duty cycles. In any case, the Residual Contact Ratio is good also for the 1% duty cycle and a 500ms beacon period, which means that most of the contact time (in this scenario the average contact time is 157s) can be effectively used for communication while keeping the energy consumption low.

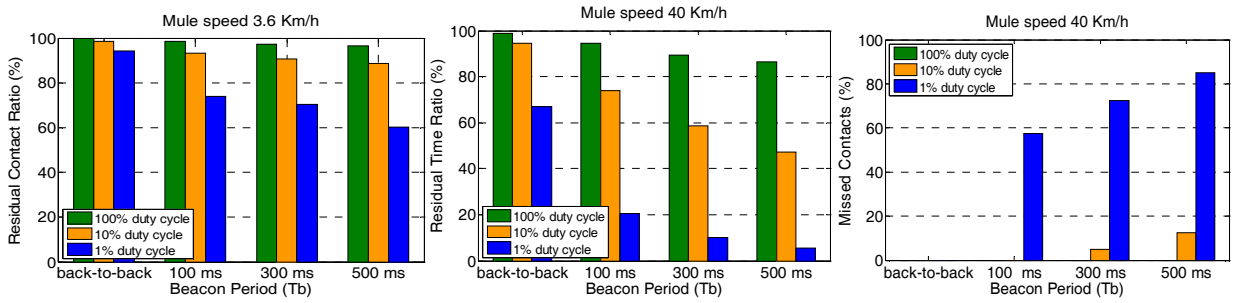


Figure 4. Residual contact ratios (a and b) and contact miss ratio (c) as a function of T_B

The results are different, instead, when the speed of the mule is high, i.e. 40 Km/h. From Figure 4-b we can see that at a 10% duty cycle, the Residual Contact Ratio drops sharply with respect to the 100% case. The situation is even worse for the 1% duty cycle where, for a beacon period of 500ms, the corresponding Residual Contact Ratio is only about 5%. Intuitively, this behavior can be explained as follows. For the same beacon period, a lower duty cycle involves longer delays in the detection of the mule. When the mule moves fast, the time it remains in the contact area is lower (in this scenario the average contact time is 18s) and, hence, a late discovery results in a small amount of time left for data transfer.

In addition, there are cases where the passage of the mule is not detected at all (i.e. contact miss). This happens because, when the contact time is short, the number of beacons emitted by the mule when it is within the contact area is low, so that the static sensor has a low probability to correctly receive a beacon. We measured the fraction of mule passages which are not detected by the static node. We found that in the low-mobility scenario (i.e., 3.6 Km/h) there is no contact miss, irrespective of the adopted duty cycle. In the 20 Km/h scenario, instead, the static node experience a limited number of contact misses, although this happens only at the 1% duty cycle. Finally, in the 40 Km/h scenario, the static node misses contacts even with a 10% duty cycle (Figure 4-c). However, this issue is limited to beacon periods above 100ms, and the resulting number of contact misses is rather low (about 12% in the worst case). On the other hand, the static node experiences a Contact Miss Ratio over 50% at the 1% duty cycle for all T_B values (excluding the unpractical back-to-back case).

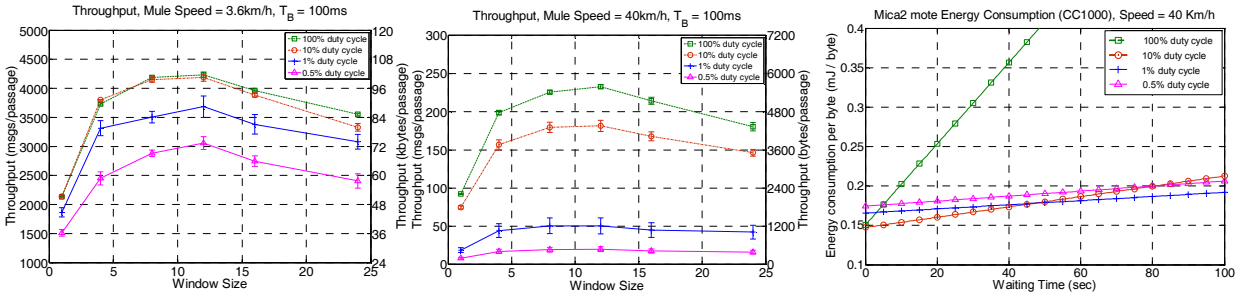
From the above experiments two main points clearly emerge. First, the beacon period should be reduced as much as possible, e.g. the minimum value feasible in the actual implementation. Second, we found that when the mule moves fast (e.g. 40 Km/h), low duty cycles result in a low Residual Contact Ratio and a large fraction of missed contacts. The main point here is if a low duty cycle can be actually used in spite of such results. To clarify this point we have to consider the data transfer phase which starts immediately after the mule discovery. We present the related results in the following subsection.

4.2 Data transfer phase

In all subsequent experiments we set the beacon period to 100ms, which has proven to be workable and efficient, based on the previous simulation results. We start considering the throughput achieved by the static node (in terms of bytes/messages per mule passage) as a function of the window size W, and for different duty cycles.

Figure 5-a shows the throughput for the 3.6 Km/h scenario. First, we can see that, irrespective of the actual duty cycle, a better performance is obtained when the window size is around 12 messages. This can be explained in this way. A small window clearly reduces the throughput, as the overhead of the acknowledgement is significant with respect to actually transmitted data. On the other hand, a large window size implies the retransmission of a lot of messages if the acknowledgement is lost. In addition, it may defer the acknowledgement to a instant in which the mule has already exited the contact area. In this case, the static node actually wastes its energy because it is transmitting data which cannot reach the mule any more.

From the same figure, we can also see that switching from a 100% to a 10% duty cycle does not impact significantly the achievable throughput, which in both cases is of about 100 KB per passage. This happens because the Residual Contact Ratios are similar for these duty cycles, so that they both allow a reasonable amount of time for data transfer. Also the 1% duty cycle exhibits good performance, as the shorter Residual Contact Time results in a maximum throughput of about 85 KB/passage, which is



a) b) c)

Figure 5. Throughput (a and b) and energy consumption per byte (c)

significant and suitable for a wide range of sensor network applications. For the sake of completeness, we have also included a 0.5% duty cycle. Also in this case a good 73 KB/passage throughput is reached as well, so that it can be actually used in some applications.

Figure 5-b shows the throughput for the 40 Km/h scenario. As in the previous case the maximum throughput is achieved for intermediate values of the window size within the considered range (i.e. about 12 messages). With respect to the previous case, we can see that there is a higher difference between values obtained with a 100% and a 10% duty cycles, i.e. 5.5 KB/passage and 4 KB/passage respectively, which are rather high. For the 1% duty cycle, instead, we get about a 1 KB/passage maximum throughput. Although this value is not so high, actually it is achieved despite a contact miss rate of nearly 60% (Figure 5-b). This result shows that a non negligible amount of data can be transferred in this case as well, and it may be enough for certain applications requiring a long network lifetime, even if we observed a large amount of contacts is missed. Even a 0.5% duty cycle might be an option for low-rate applications where an achievable throughput of 400 bytes/passage is enough.

4.3 Energy expenditure

In the previous sections we have discussed the performance of the data collection process, in terms of both discovery and data transfer efficiency. In this section we want to evaluate the average energy consumed by the static sensor node for each byte (message) correctly transferred to the data mule. To this end, as the discovery phase generally does not start as soon as the mule enters the contact area, we have to consider the waiting time as well (see Figure 3), which is caused by the uncertainness of the mule arrival time. In all the experiments discussed below we set the beacon period to 100ms and the window size to 12 messages.

Figure 5-c shows the energy consumption per successfully transferred message/byte as a function of the average waiting time, when the data mule moves at a speed of 40 Km/h. We can see that a lower duty cycle does not necessarily imply a lower unitary energy consumption. In fact, there is a range of waiting times in which a low duty cycle (i.e. 0.5% and 1%) is less energy efficient than a higher duty cycle (i.e. 10% and even 100%). This counter-intuitive behavior can be explained by observing that in this scenario the contact time is limited due to speed of the data mule. When the average waiting time is low (i.e., the mule mobility pattern is somewhat predictable so that the sensor can wake up just before the mule is expected to arrive) it is more convenient to use a large duty cycle, which allows a timely mule discovery and a large residual contact time, rather than a low duty cycle which misses the contact most of the time and, in any case, allows a short residual contact time. Furthermore, the 0.5% duty cycle always has a higher energy expenditure than the 1% duty cycle. Although this result may sound strange at first, actually it can be easily explained. As the contact time is short in this scenario, most of the energy is consumed by the static node when it is in duty cycle mode. Passing from a 1% to a 0.5% duty cycle thus approximately halves the energy consumption. However, we can see from Figure 5-b that the corresponding throughput drops from 1.2 Kbytes to 480 bytes, i.e., it is more than halved. In other words, the energy saving is not compensated by the throughput decrease.

The same is not true when the speed is 3.6 Km/h, as the energy consumption per byte (message) increases linearly with the waiting time and decreases with the duty cycle (we have omitted the figure for the sake of space). This is because, when the mule moves slowly, there is always enough time available for data transfer, even when the duty cycle is low (e.g. 1%). Therefore, using a lower duty cycle reduces the unitary energy consumption. In particular, it emerges that, in this case, using a duty cycle larger than 10% is never convenient.

5. Conclusions

In this paper we have analyzed the problem of reliable and energy-efficient data collection in sensor networks with data mules. We have provided an integrated evaluation of mule discovery and data transfer performance. Our simulation results show that a discovery protocol with a low duty cycle (e.g., 1%) is very energy efficient, while allowing at the same time a throughput actually satisfactory for common environmental monitoring applications based on wireless sensor networks. Furthermore, we found that – depending on the mule mobility pattern – a low duty cycle might not be always the most convenient option from the energy efficiency point of view. This is especially true when mule arrival times can be somewhat predicted, notwithstanding some uncertainness. This clearly points out that the joint effect of mule discovery and data transfer has to be carefully considered in the design and deployment stages of the network, in order to prevent wrong assumptions leading to energy wastage.

These results pave the way for directions involving more sophisticate wakeup strategies. In other words, the sensor might wake up less frequently (i.e. intentionally missing a number of passages) but operate at a higher duty cycle during discovery. This choice might be more convenient than waking up at each expected mule arrival, but operating with a lower duty cycle. To this end, the results provided in this paper could be used as a base for defining adaptive discovery and data transfer protocols. The goal of these protocols should be the automatic selection of the operating parameters (e.g., the duration of the sleeping phase, the duty cycle for discovery, the size of the data transfer window, and so on) based on the actual state of the sensor node (e.g. buffer level, time elapsed from the last contact, residual energy, etc.). We are currently investigating such adaptive schemes.

References

[1] G. Anastasi, M. Conti, E. Gregori, C. Spagoni, and G. Valente, "Motes Sensor Networks in Dynamic Scenarios", *International Journal of Ubiquitous Computing and Intelligence,* Vol. 1, N. 1, April 2007.

[2] G. Anastasi, M. Conti, E. Monaldi, and A. Passarella, "An Adaptive Data-transfer Protocol for Sensor Networks with Data Mules", *Proc. IEEE WoWMoM 2007*, Helsinki (Finland), June 18-21, 2007.

[3] L. Bölöni and D. Turgut, "Should I Send Now or Send Later? A Decision-theoretic Approach to Transmission Scheduling in Sensor Networks with Mobile Sinks", *Wireless Communications and Mobile Computing Journal (WCMC)*, 2007.

[4] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks", *Proc. IPSN 2000*, Palo Alto, USA, 2003.

[5] CrossBow Technology Wireless Sensor Networks, Website: `http://www.xbow.com`

[6] Y. Gu, D. Bozdag, E. Ekici, F. Ozguner, and C. Lee, "Partitioning Based Mobile Element Scheduling in Wireless Sensor Networks", *Proc. IEEE SECON 2005*, pp. 386- 395, S. Clara, USA, Sept. 2005.

[7] Y. Gu, D. Bozdag, and E. Ekici, "Mobile Element Based Differentiated Message Delivery in Wireless Sensor Networks", *Proc. IEEE WoWMoM 2006*, Niagara Falls, USA, June 2006.

[8] S. Jain, R. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks", *ACM/Springer Mobile Networks and Applications*, Vol. 11, No. 3, June 2006, pp. 327-339.

[9] D. Jea, A. Somasundra, and M. Srivastava, "Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks", *Proc. IEEE DCOSS 2005*, Marina del Rey, USA, 2005.

[10] H. Jun, M. Ammar, and E. Zegura, "Power Management in Delay Tolerant Networks: A Framework and Knowledge-Based Mechanisms", *Proc. IEEE Conference on Sensor and Ad Hoc Communication and Networks (SeCon 2005)*, September 2005.

[11] A. Kansal, A. Somasundara, D. Jea, M. Srivastava, and D. Estrin, "Intelligent Fluid Infrastructure for Embedded Networks", *Proc. ACM Mobisys 2004*, Boston, USA, June 2004.

[12] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications*", Proc. of SenSys '03*, Los Angeles, California, USA, November 05 - 07, 2003.

[13] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring", *Proc. ACM WSNA 2002*, pp 88-97 Atlanta, USA, Sept. 2002.

[14] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks, *Proc. IEEE SNPA 2003,* May 2003, pp. 30- 41.

[15] A. Somasundara, A. Kansal, D. Jea, D. Estrin, and M. Srivastava, "Controllably Mobile Infrastructure for Low Energy Embedded Networks", *IEEE Transactions on Mobile Computing*, Vol. 5, N. 8, August 2006.