An Adaptive Data-transfer Protocol for Sensor Networks with Data Mules

Giuseppe Anastasi[*], Marco Conti[#], Emmanuele Monaldi[*], Andrea Passarella[#]

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

In this paper we deal with energy-efficient data collection in sparse sensor networks with data mules. We analyze the problem of optimal data transfer from sensors to data mules, and derive an upper bound for the performance of ARQ-based data-transfer protocols. This analysis shows that protocols currently used have low performance, which results in unnecessary energy consumption. Based on these results we define and evaluate an Adaptive Data Transfer (ADT) protocol that is able to combine efficiency and adaptability to external conditions. Simulation results show that ADT not only reduces significantly the average data-transfer time in comparison with previous protocols, but also provides quasi-optimal performance. In addition, it is able to react quickly to variations in the external conditions and adapt to new conditions in a limited time.

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 [9]. 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 condition 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.

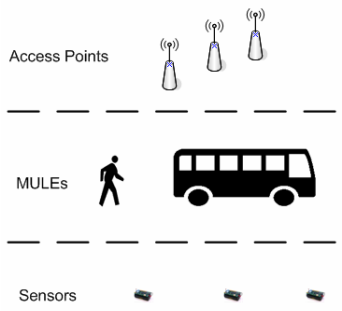

Figure 1. Data Mule Architecture.

Data collection in such sensor networks can be achieved more efficiently by using data mules (or mules for short), i.e., mobile elements that carry data from static sensors to an infra-structured AP [5] (see Figure 1). Depending on the application scenario, mules may be either part of the external environment (e.g., buses, cabs, or walking people), or part of the network infrastructure (e.g., mobile robots). They visit static sensors at predictable or random times (depending on their nature and mobility pattern), pick up data, and carry them to an AP.

Mules are assumed to be power renewable, while static sensors 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 sensor transfers its data to the mule) must be energy efficient to prolong the lifetime of sensors. 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. The design of the mule discovery protocol is beyond the scope of this paper. Here we just assume that the discovery protocol allows a timely mule detection. This is usually achieved by letting the mule broadcast beacon messages periodically. The sensor is typically on a low duty cycle while waiting for the mule, and switches the radio

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 and PRIN WiseMaP projects.

to the fully operational mode as soon as it receives a beacon [5, 11].

In this paper we focus on the energy efficiency of the data-transfer protocol. A major contribution of this work is the analysis of the optimal ARQ-based data-transfer protocol, which provides an upper bound for any ARQ-based data-transfer protocol. We show that currently used protocols are very simple, at the cost of low performance in terms of energy consumption (expressed as duration of the data transfer). Based on this result, we propose the Adaptive Data Transfer (ADT) protocol that reduces significantly the time required by a sensor to transfer its messages to the mule, performing remarkably close to the optimal case. At the same time, ADT is able to adapt to variations in the external conditions.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 analyzes the problem of optimal data transfer in sensor networks with data mules. Section 4 is devoted to the ADT protocol description and evaluation. Section 5 concludes the paper.

2. Related Work

The bibliography on wireless sensor networks is extremely large. However, we focus here on sensor networks with mobile elements for data collection. The data mule model was first proposed independently in [2] and [10] to address the problem of energy-efficient data collection in sparse sensor networks. In [5] the model is evaluated by analysis and simulation. The authors investigate the impact on the data success rate, latency, and energy cost of a large set of operating parameters (e.g., data generation rate, sensor buffer size, sensor duty cycle, mule inter-arrival distribution, bit rate, transmission range). In certain application scenarios multiple mules may be required to meet performance requirements. In [6] the authors investigate the benefits of using load balancing techniques to assign sensors to mules when multiple mules are used.

A key ingredient for energy-efficient data collection is the communication protocol used for transferring data from the sensor to the mule. A protocol is proposed in [2] which relies on the assumption of circular transmission range, negligible message loss rate within the transmission range, and predictable mule arrival times. This communication model is clearly not realistic. The experimental analysis in [1] has shown that, as expected, the transmission range is not circular. In addition, the message loss probability depends on the mule position, and – at a very fine granularity – does not decrease monotonically with the sensor-mule distance. Nevertheless, – at a coarser time granularity – the message loss can be assumed as a monotonically decreasing function. Specifically, when the mule is within the transmission range of the sensor, but very far from it, the message loss probability may be so high to make the available throughput extremely low. On the other hand, when the mule is close to the sensor the message loss probability becomes very low and the communication very efficient. The ADT protocol proposed in this paper is based on the lesson learned from the experimental analysis in [1].

A simple *stop-and-wait* data transfer protocol is used in [7, 11]. The static sensor starts transmitting as soon as it discovers the mule in its proximity. No information about the mule location is exploited because such information may not be available in all systems. And, when available, it may be unreliable due to variations in the wireless channel, multi-path effects, and inaccuracies in the estimation procedure. The ADT protocol uses an ARQ-based communication scheme like the protocol in [11]. However, its design leverages the analytical study of the optimal case also provided in this paper. With respect to a stop-and-wait scheme, we show that a window size larger than one increases the available throughput and reduces the duration of the data transfer (and, thus, energy consumption). The ADT protocol too does not rely on information about the mule position. However, the sensor tries to guess the time instant when the mule will be at the minimum distance from it. Then, it transmits its data around the expected minimum-distance point. Clearly, this allows ADT to exploit better (and, usually, the best) channel conditions (see below for details).

An important issue in the design of a data transfer protocol for sensor networks with mules is the mule's behavior. In fact, it is important to know the mule inter-arrival distribution, and whether or not the mule's motion can be controlled in some way. In [2] the authors assume predictable mule arrival times. This simplifies the discovery process and helps scheduling data transmissions. Other papers assume that the mule speed can be controlled by the mule itself [11]. Finally, when sensor nodes have different data generation rate some sensors may need to be visited more frequently than others. To face this problem several scheduling algorithms have been proposed [3, 4, 11]. The basic idea is to schedule the visits of the mule to sensors in such a way to avoid buffer overflows at sensors. In this paper, to make the analysis as general as possible, we assume random mule arrival times, and no control on the mule's motion. We do not address the problem of mule's movements scheduling as we focus on the data transfer phase.

3. Optimal Data Transfer

In this section we analyze the problem of data transfer from an ideal point of view. Specifically, we define an optimal ARQ-based data-transfer protocol that minimizes energy consumption at the sensor node, and evaluate its performance. The purpose of this analysis is to derive an upper bound for the performance of ARQ-based protocols. To simplify the analysis, but without losing in generality, in the following we will refer to the scenario shown in Figure 2. We consider a single static sensor and a single mule moving along a linear path at a fixed vertical distance (D_y) from the sensor (this is a reasonable assumption if we consider that the contact time between the sensor and the mule is typically short). In addition, we will assume a constant mule's speed. Throughout we will consider the horizontal distance D_x as negative when the mule is approaching the static sensor, and positive in the reverse direction. Under the assumption of constant speed v, the time needed by the mule to cover a distance D_x is given by $t_x = \frac{D_x}{v}$.

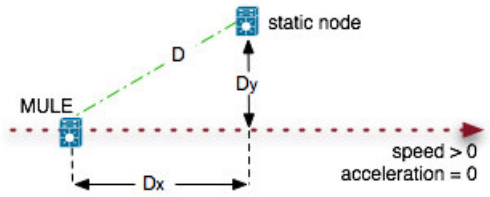

Figure 2. Reference scenario.

3.1 Problem formulation

We start our analysis from the evidence that in a real environment the message loss has a very irregular behavior. As shown by the experimental data in Figure 3 [1], the message loss probability decreases with the sensor-mule distance, (but not monotonically), and, in general, it is not symmetric with respect to the minimum-distance point, i.e., D_x=0 (even though it can be reasonable assumed as approximately symmetric in many practical cases).

Intuitively, to minimize the data transfer time (and, hence, energy consumption) the sensor should transmit its data in the time interval with minimum message loss probability, i.e., around the minimum-distance point. More formally, the optimal data-transfer problem can be stated as follows. Let $Th(t)$ denote the instantaneous throughput available on the wireless link at time t, and B the amount of data (in number of messages) to be transferred from the sensor to the mule during the contact. To minimize the data transfer time we need to find the minimum time interval (t_1, t_2) that allows the correct transmission of all B messages. This can be formalized as follows:

$$\begin{aligned} &\text{minimize} && t_2 - t_1 \\ &\text{subject to} && \int_{t_1}^{t_2} Th(t) \cdot dt = B \end{aligned} \qquad (1)$$

where $\int_{t_1}^{t_2} Th(t) \cdot dt$ is the amount of data than can be transferred in the time interval (t_1, t_2).

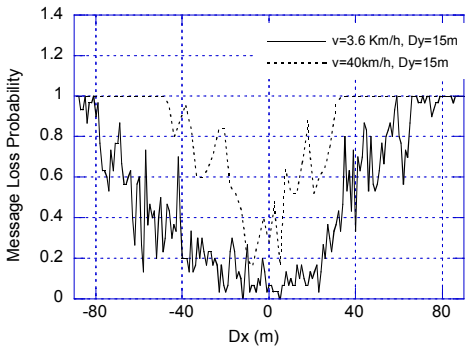

Figure 3. Message loss behavior in a real environment.

3.2 Throughput Model

The available throughput $Th(t)$ in (1) depends on the specific communication protocol that is used. We consider here an ARQ-based scheme with window size w and selective retransmission[1]. In such a protocol the sensor transmits w consecutive fixed-size data messages and, then, stops waiting for an *ack* from the mule. The ack informs that the mule is still within the communication range, and also notifies which messages have been received correctly. The subsequent window is thus used by the sensor partly for re-transmitting messages lost in the previous round, and partly for transmitting new messages. To derive the available throughput we will assume in the following that the time is slotted and each communication window consists of w time slots for message transmissions from the sensor to the mule, and one more slot for ack transmission in the reverse direction (see Figure 4). The following notations will be used throughout:

δ Maximum time required to transmit a message or an ack (slot size);

T_w duration of a communication window including

[1] A similar protocol with $w=1$ has been used in [11]. In that protocol the sensor starts transmitting messages as soon as it discovers the data mule in its communication range.

the ack message ($T_w = (w+1)\cdot\delta$);

$N_i(t)$ number of messages successfully transferred in the $i-th$ slot of the window centered at time t;

$N(t)$ number of messages successfully transferred in the entire window centered at time t;

$p(t)$ loss probability experienced by a message transmitted by the sensor node at time t;

$q(t)$ loss probability experienced by an ack or beacon message transmitted by the mobile mule at time t.

Figure 4. Window around time *t*.

Let us focus on a single window of size w centered around a generic time t, as shown in Figure 4 (we assume that w is an even number). Given the small duration of each single slot we also assume that the loss probability is constant within the slot. Instead, messages transmitted in different slots within the same window will experience different loss probabilities, i.e.,

$$p\left(t-\frac{w}{2}\delta\right),\cdots,\ p(t-\delta), p(t), p(t+\delta),\cdots, p\left(t+(\frac{w}{2}-1)\delta\right)$$

The ack message will experience a loss probability given by $q\left(t+\frac{w}{2}\delta\right)$.

From the sensor's point of view, a message transmitted during the slot starting at time $t+i\delta$ (where i is an integer number in the range $\left[-\frac{w}{2},\frac{w}{2}-1\right]$), is assumed to be *transferred* to the mule if the message is received correctly by the mule, *and* the related ack is received successfully by the sensor, i.e., with probability $[1-p(t+i\delta)]\cdot\left[1-q\left(t+\frac{w}{2}\delta\right)\right]$.

The number $N_i(t)$ of messages successfully transferred in the generic time slots is thus

$$N_i(t)=\begin{cases}1 & with\,prob\ [1-p(t+i\delta)]\cdot\left[1-q\left(t+\frac{w}{2}\delta\right)\right]\\ 0 & with\,prob\ 1-\left\{[1-p(t+i\delta)]\cdot\left[1-q\left(t+\frac{w}{2}\delta\right)\right]\right\}\end{cases} \quad (2)$$

And the number $N(t)$ of messages transferred in the whole window centered at time t is given by:

$$N(t)=\sum_{i=-\frac{w}{2}}^{\frac{w}{2}-1}N_i(t) \quad (3)$$

Hence,

$$E[N(t)]=E\left[\sum_{i=-\frac{w}{2}}^{\frac{w}{2}-1}N_i(t)\right]= \quad (4)$$

$$=\left[1-q\left(t+\frac{w}{2}\delta\right)\right]\cdot\sum_{i=-\frac{w}{2}}^{\frac{w}{2}-1}[1-p(t+i\delta)]$$

From Equation (4) we can derive the instantaneous throughput $Th(t)$. For simplicity we approximate $Th(t)$ with the throughput achieved in the communication window centered at time t, i.e., in the time interval $\left[t-\frac{w}{2}\delta, t+\left(\frac{w}{2}+1\right)\delta\right]$:

$$Th(t)=\frac{E[N(t)]}{T_w}=\frac{E[N(t)]}{(w+1)\cdot\delta} \quad (5)$$

Hence, equation (1) can be re-written as:

$$\text{minimize}\quad t_2-t_1$$

$$\text{subject to}\quad \int_{t_1}^{t_2}\frac{\left[1-q\left(t+\frac{w}{2}\delta\right)\right]\cdot\sum_{i=-\frac{w}{2}}^{\frac{w}{2}-1}[1-p(t+i\delta)]}{(w+1)\cdot\delta}\cdot dt=B \quad (6)$$

Real message loss functions $p(t)$ have a very irregular behavior (see Figure 3). To simplify our analysis we derived an analytically-tractable loss model on the basis of the message losses measured in [1]. By using the least square interpolation method, we derived a polynomial interpolation of real probability loss functions. Then, we used simulation to assess the accuracy of our loss model, and decide the degree of the polynomial function. We compared the performance of an ARQ-based data-transfer protocol, like the one described above, when using the real loss curve and polynomial model, respectively. The comparison was done in terms of percentage of messages successfully transferred from the sensor to the mule, for different number of messages to transfer (*B)*. We found that a 2-degree polynomial model is precise enough for our purposes, as it provides a sufficient accuracy while keeping the complexity low. Therefore, hereafter we will assume

$$p(t)=a_2\cdot t^2+a_1\cdot t+a_0 \quad (7)$$

where coefficients a_0, a_1, a_2 depend on the specific scenario (i.e., mule's speed v, distance D_y), and are reported in Table 1 for the case D_y=15 m.

Table 1. Coefficient values for different mule's speeds (D_y=15 m).

Coefficient	v=3.6 Km/h	v=20 Km/h	v=40 Km/h
a_0	0.133	0.364	0.405
a_1 (s^{-1})	0	0	0
a_2 (s^{-2})	0.000138	0.0109	0.0502

3.3 Performance Analysis

To assess the effectiveness of the optimal data-transfer approach, in this section we compare the performance of two ARQ-based protocols using different approaches for choosing the transmission interval. The first protocol, throughout referred to as *optimal protocol*, follows the optimal approach and selects the transmission interval according to equation (6). In the second protocol, throughout referred to as *naive protocol*, the sensor node starts transmitting as soon it detects the data mule within its communication range. Protocols following the naïve approach are largely used in literature due to their simplicity [2, 11]. The data transfer time for the naïve protocol is still an interval (t_1, t_2) such that

$$\int_{t_1}^{t_2} \frac{\left[1-q\left(t+\frac{w}{2}\delta\right)\right]\cdot\sum_{i=-\frac{w}{2}}^{\frac{w}{2}-1}\left[1-p(t+i\delta)\right]}{(w+1)\cdot\delta}\cdot dt = B \qquad (8)$$

However, now t_1 coincides with the time instant when the sensor node discovers the mule, and the interval (t_1, t_2) is not, in general, the minimum one.

Table 2. General Parameter Settings

Parameter	Value
Vertical distance (D_y)	15 m
Bit rate	19.2 kbps
Message size	20 bytes
Frame size (data + control information)	36 bytes
time slot (δ)	50 msec

In the following analysis we will use the parameter values in Table 2 (which refer to the Mote platform) and assume $p(t)=q(t)$, for any t. As a preliminary step, we validate by simulation the analytical models (6) and (8). To this end, we implemented the above data transfer protocols in the TOSSIM simulation tool [8], and ran several experiments by using the replication method with a confidence level of 90%. Message-loss probabilities were generated according to the polynomial model (7).

Figure 5 shows the average data-transfer time as a function of the window size when the mule's speed is 40 Km/h and B=10 messages. We also performed other experiments with different values for B and v (3.6 and 20 Km/h). The results are not reported here for the sake of space, but they are (qualitatively) similar to those in Figure 5. For both protocols, there is a strong agreement between analytical and simulation results. Therefore, in the remaining part of this section we will refer to analytical results only.

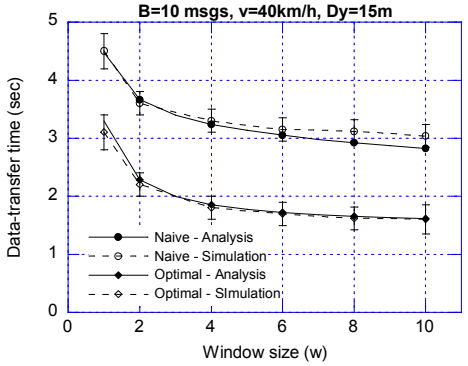

Figure 5. Analytical and simulation results when the mule speed is 40 km/h.

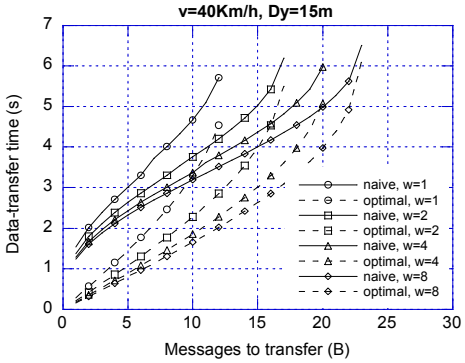

Figure 6. Average transfer times for different B values with the optimal and naïve protocols.

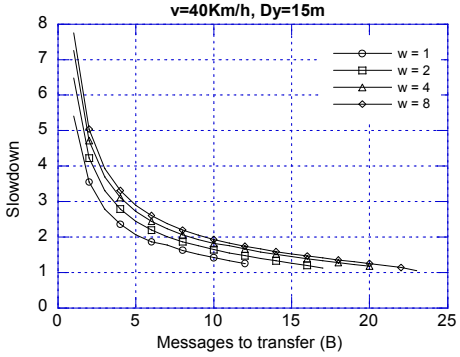

Figure 7. Slowdown in the average transfer time provided by the optimal protocol.

Figure 6 compares the average times required by the optimal and naïve protocols for the reliable transfer of the same amount of data. As expected, the average

data-transfer time is always much shorter with the optimal protocol. For a given B value, the difference is in the order of seconds (tens of seconds when the mule's speed is 3.6 Km/h). The same behavior is also highlighted in Figure 7 showing the slowdown, i.e., the ratio between the average transfer time of the naïve and optimal protocols. The slowdown may be up to several times (even 20 when the mule's speed is 3.6 Km/h) especially when the amount of data to transfer is small. In such a case the transfer time is very short compared with the contact time and, hence, the optimal protocol can transmit when the message loss is very low. When the required transfer time is comparable with the contact time, the intervals selected by the optimal and naive protocols tend to overlap, and both protocols exhibit similar performance.

From the above figures we can also observe that the window size has a significant impact on the average duration of the data transfer. Intuitively, a larger window size reduces the overhead related to the acknowledgement and, thus, increases the available throughput. On the other hand, a very large window size may not be appropriate as the purpose of acks in this context is not acknowledging the correct reception of data messages by the mule, but also notifying the sensor that the mule is still within the communication range.

4. Adaptive Data Transfer Protocol

4.1 Rationale

The optimal protocol outperforms significantly the naïve protocol. However, it is unpractical because all the assumptions it relies upon are rarely (or never) met in practice. First, the message loss probability may not be available. Even if available, it is subject to frequent changes due to variations in temperature, humidity, meteorological conditions, and so on. Finally, the real contact time may be shorter than that derived theoretically from the message loss function. Sensors typically operate on a low duty cycle while waiting for the mule arrival and, hence, they could miss some initial beacons.

On the other hand, the naïve approach is practical and robust even if it provides very low performance. In this section we try to combine the efficiency of the optimal approach and the robustness of the naïve approach by designing a new protocol called *Adaptive Data-Transfer* (ADT) protocol. Like the optimal and naïve protocols, it uses an ARQ scheme for data transfer.

The main assumption of the ADT protocol is that the minimum message loss between the mule and the sensor occurs at the minimum distance point, and that the mule's passage on this point occurs exactly at mid contact time. ADT also assumes that the message loss curve is approximately symmetric with respect to the mid contact point. Based on these assumptions, ADT transfers the messages during a symmetric time interval centered on the (estimated) mid contact point, which approximates the optimal time interval (t_1,t_2) derived by the optimal (unfeasible) protocol.

4.2 Protocol Description

The ADT protocol includes a *startup phase* and a *steady phase*. The startup phase provides an estimate of the contact time, i.e., the time interval during which the mule will remain in the communication range of the static sensor. This information will then be used to optimize the data transfer process during the steady phase. The two phases could be partially overlapped. However, for simplicity, in the following we will assume them as completely separated.

The startup phase spans one or more mule's passages. At each passage the sensor measures the time interval between the reception of the first and last beacons from the mule. These measures are then used to derive an estimate of the contact time. As the external condition may vary over time, this estimate is updated periodically during the steady phase (every T_{CT} mule's passages), as follows

$$\hat{CT}(n+1) = \alpha_{CT} CT(n) + (1-\alpha_{CT}) \cdot \hat{CT}(n) \qquad (9)$$

where $\hat{CT}(n)$ $(CT(n))$ is the contact time *estimated* (*measured*) at round n, and α_{CT} a real value in [0,1].

In the steady phase the ADT protocol works as follows. Initially, the sensor is on a low duty cycle to save energy. Upon receiving a beacon from the mule it switches to the fully operational mode, and estimates the expected data transfer time (*DTT*), i.e., the time needed to transfer all the messages in the buffer. To simplify the description we assume here that the amount of data to transfer is the same in all successive passages. At the first passage the expected *DTT* is conservatively taken equal to the estimated contact time. In subsequent passages, it is calculated on the basis of actual *DTTs* measured in previous rounds, as

$$\hat{DT}(m+1) = \alpha_{DTT} DTT(m) + (1-\alpha_{DTT}) \cdot \hat{DTT}(m) \quad (10)$$

where $\hat{DTT}(m)$ $(DTT(m))$ is the data transfer time *estimated* (*measured*) at round *m,* and α_{DTT} a real value between 0 and 1. However, if in the previous passage it was not possible to transfer *all* the messages in the buffer, $\hat{DTT} = \hat{CT}$ is used. Assuming that the reception time of the first beacon is used as the time origin, the data transfer should start after a waiting time

$WT = \frac{\hat{CT} - \hat{DTT}}{2}$. If WT is greater than the sum of the delays required by the radio to transition from the active to the sleep mode (T_OFF) and back again (T_ON), the sensor puts the radio in sleep mode for a time $(WT - T\_ON)$ to save energy

The next step consists in transferring data to the mule. The sensor node transmits back to back a number of messages equal to the window size (W_SIZE) and, then, stops waiting for an ack from the mule. If the ack is not received within a pre-defined timeout, the sensor increases a counter (*missed_acks*), and retransmits all messages. After ACK_MAX consecutive missed acks the sensor assumes that the mule has exited the communication range and stops the data transfer. If the ack is received on time the *missed_acks* counter is reset. Each ack includes a bitmap specifying messages received correctly by the mule. Such messages are thus removed from the buffer, while the other messages will be retransmitted in the next window. The process goes on until all messages have been transmitted, or ACK_MAX consecutive missed acks have been detected, or the estimated residual contact time is less than T_w. If all messages in the buffer have been transferred the sensor calculates the total DTT (difference between the initial and final transmission times) that will be used to derive $\hat{DTT}$ at the next round.

4.3 Performance Evaluation

To compare the performance of ADT with those of the naïve and optimal protocols, we implemented ADT in the TOSSIM simulator, and ran a set of experiments using the parameter settings shown in Table 2 and Table 3. The message loss probability was generated according to the polynomial model derived above[2].

Table 3. ADT Parameter Settings.

T_ON=T_OFF=1.5 ms	W_SIZE=8
T_{CT}=10	*missed_acks*=3
α_{CT} =0.8	α_{DTT} =0.5

Figure 8 compares the temporal behavior of the three protocols (the startup phase of ADT is omitted). In the first part of the experiment (until the 50th passage) the mule moves at a speed of 40 Km/h, and the optimal protocol is perfectly tuned to the external message loss conditions. Specifically, it uses the polynomial model (7) with appropriate coefficients (those related to 40 Km/h in Table 1) to predict the data transfer time and decide the initial transmission instant. The naïve protocol requires the largest data-transfer time because it transmits in the initial part of the contact time, when the message loss rate is high. ADT initially performs as the naïve protocol as it cannot rely on information about previous data transfers. After few passages, however, ADT converges to a quasi-optimal behavior.

We also investigated how the three protocols react to changes in the external conditions. To this end we introduced some variations in the external conditions. After the 50th passage the mule's speed changes from 40 Km/h to 20 Km/h (the message loss curve and contact time vary accordingly). Finally, after the 100th passage the speed changes back to 40 Km/h. Figure 8 shows how each of the three protocols reacts to such variations. When v decreases from 40 to 20 Km/h, the data transfer time of the naïve protocol increases because the contact time is now longer and the sensor starts transmitting when the mule is at a higher distance than before. Therefore, it experiences a greater loss rate. The optimal protocol is not aware of the variation and, thus, remains tuned to the previous conditions (i.e., data transfer times are still predicted according to the message loss model related to 40 Km/h). This results in increased data-transfer times. The ADT behavior is more complex to analyze. Just after the variation has occurred, ADT behaves close to the wrongly-tuned optimal protocol. This is because the estimate of the contact time is recalculated every T_{CT} mule's passages. In the experiment shown in Figure 8 T_{CT} is set to 10, and the variation in the mule's speed occurs just after the contact time has been recalculated. Under these conditions the protocol realizes that the contact time has changed only after T_{CT} passages (*reaction latency*). In general, the duration of the reaction latency is, on average, equal to half T_{CT}. After the reaction latency the protocol progressively adapts to new conditions and reduces the data transfer time towards a quasi-optimal steady-state value. A significant decrease in the data-transfer time can be observed at every T_{CT} passages (e.g., 60, 70, 80). This is because at these passages the contact time is recalculated and, thus, a more accurate estimate is available. Finally, when the mule's speed is set back to 40 Km/h, both the naïve and optimal protocols immediately switch back to the initial behavior. The ADT protocol experiences the reaction latency and a subsequent transient phase towards the quasi-optimal behavior. From Figure 8 it clearly emerges that the average data-transfer delay required by ADT is considerably lower than that of the naïve protocol. Also, ADT is auto-tuning, and always tends to a quasi-

[2] As above, we used the replication method with 90% confidence level to derive confidence intervals, and assumed $p(t) = q(t)$.

optimal behavior under varying external conditions. Thus, ADT is a valid approximation of the optimal (but unfeasible) protocol.

Finally, we have also measured the length of the initial transient phase required by ADT to learn the data-transfer time from scratch. This interval depends on a number of parameters such as contact time (i.e., mule's speed), number of messages to transfer (*B)*, α_{DTT} value. In our experiments we measured the initial transient phase as the interval between the start of the steady phase and the time when the fluctuations of the data-transfer time are less than 10% of the steady-state value. Table 4 shows the average duration of the initial transient phase, expressed in number of mule's passages, for different parameter settings. The initial transient phase becomes longer and longer as the ratio between the data-transfer time (which depends on *B*) and the contact time (which is related to the inverse of *v*) decreases. This can be explained by observing that expected data-transfer times are obtained by (10) using the (estimated) contact time as the initial value. If the contact time is large, with respect to the data transfer time, it affects a large number of predictions, resulting in a longer transient phase.

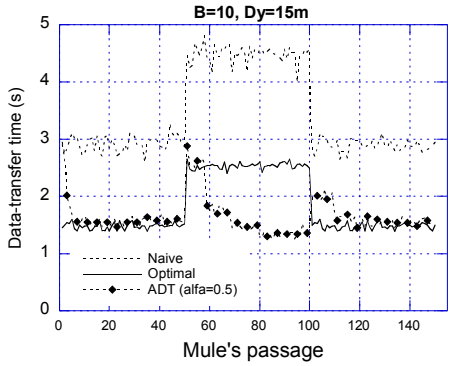

Figure 8. Behavior of the three protocols.

In all the experiments summarized in Table 4 we used $\alpha_{DTT} = 0.5$ to predict the data transfer time at each passage. By tuning α_{DTT} appropriately we can control the protocol behavior during the initial transient phase. If we use a large α_{DTT} value (e.g., $\alpha_{DTT} = 0.9$) ADT converges quickly to the optimal behavior, but successive data-transfer times have large fluctuations. On the other hand, a small α_{DTT} value (e.g., α_{DTT}*=0.1*) provides a slower convergence to the optimal behavior but, also, smaller fluctuations.

5. Conclusions

In this paper we have analyzed the problem of optimal data transfer in sensor networks with data mules. Currently used (naïve) protocols are inefficient from an energy consumption standpoint. We have proposed the ADT protocol, showing that its energy consumption is just slightly higher with respect to the optimal (but unfeasible) case. We have performed our analysis by considering sparse sensor networks where nodes are considerably far apart from each other. Currently, we are extending our protocol to manage the more general case where sensors are deployed in such a way to form local clusters (with a cluster-head in charge of collecting data from other nodes in the cluster, and transferring them to the mule), and the amount of data to transfer may be different in different passages. Also we are working on analyzing the effects of the mule discovery protocol on the data transfer protocol.

Table 4. Duration of the initial transient phase in the ADT protocol (D_y=15 m, α_{DTT}=0.5).

v	B=10	B=40	B=100
3.6 Km/h	16.6 ±2.4	12.9 ± 2.0	8.1 ± 1.2
20 Km/h	5.9 ±1.5	4.7 ± 0.9	-
40 Km/h	5.6 ± 0.9	-	-

References

[1] G. Anastasi, M. Conti, E. Gregori, C. Spagoni, G. Valente, "Motes Sensor Networks in Dynamic Scenarios", *International Journal of Ubiquitous Computing and Intelligence,* Vol. 1, N.1, January 2006.

[2] A. Chakrabarti, A. Sabharwal, B. Aazhang, "Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks", *Proc. IPSN 200)*, Palo Alto, USA, 2003.

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

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

[5] S. Jain, R. Shah, W. Brunette, G. Borriello, S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks", *ACM/Springer Mobile Networks and Applications*, Vol. 11, pp. 327-339, 2006.

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

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

[8] P. Levis, N. Lee, M. Welsh, D. Culler, "TOSSIM Accurate and Scalable Simulation of Entire TinyOS Applications", Proc. *ACM SenSys 2003*, Los Angeles, 2003.

[9] 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.

[10] 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.

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