Cinzia Cappiello, Fabio A. Schreiber

Quality- and Energy-Aware Data Compression by Aggregation in WSN Data Streams

Proceedings of the

Seventh Annual IEEE International Conference on Pervasive Computing and Communications

PerSeNS 2009 Workshop

9-13 March 2009, Galveston

pp. 634-639

Los Alamitos, California

Quality- and Energy-Aware Data Compression by Aggregation in WSN Data Streams

Cinzia Cappiello, Fabio A. Schreiber
Politecnico di Milano, Dipartimento di Elettronica e Informazione
Via Ponzio 34/5, 20133 Milano, Italy
Email: {cappiell, schreibe}@elet.polimi.it

***Abstract*—Sensor networks consist of autonomous devices that cooperatively monitor an environment. Sensors are equipped with capabilities to store information in memory, process information and communicate with neighbors and with a base station. However, due to the sensors' size, their associated resources are limited. In such a context, the main cause of energy dissipation is the use of the wireless link. Solutions that minimize communication are needed. In this paper a framework to manage efficiently data streams is presented. The proposed approach aims at saving energy by capturing signals and compress them instead of sending them in raw form. The algorithm also guarantees that the compressed representation satisfies quality requirements specified in terms of accuracy, precision, and timeliness.**

I. Introduction

Owing to the rapid advancement of semiconductor manufacturing technology, electronic devices are getting smaller, cheaper, and requiring less power for their operation every year. Wireless Sensor Networks (WSNs) have greatly profited of this technological advancement. Currently, sensors are getting more complex and they must be treated as equal partners in future distributed database systems as they can store, manipulate and communicate information. In fact, each sensor produces a continuous stream of data which flows from the sensor node itself to the consumer node - usually one or more base stations - possibly by multi-hop transmission.

However, the size of each sensor node is expected to be small and this size constraint implies limitations on the associated resources. In fact, when dealing with wireless sensors, the designer faces two main technological issues, such as memory and power amount. On the one hand, memory can only host few, possibly compacted, measurement data for a limited time span and it is necessary to periodically transfer data to a large storage device. On the other hand, the life of on-board batteries is limited and transmission is the most power consuming function. These constraints conflict with each other since the need of transmitting data in order to free the sensor local memory requires frequent transmissions of long data sequences which are highly power consuming. Thus, we need to compact incoming data in order to optimize the local storage need and transmit only few value-added data to the parent nodes. Obviously, since sending data is more power consuming than processing them, limiting communication in a sensor network will benefit all involved parties. Data compression is one effective method to use limited resources of WSNs. It is assumed that some loss of precision in the compressed version of incoming data can be tolerated if this helps in reducing communication. However, it is important that some data quality requirements are satisfied and the error introduced in the compression is below a specified threshold.

Data compression algorithms have been thoroughly studied for years. Traditionally, data compression approaches focused on saving storage and not energy. As a result, the compression ratio is their fundamental metric. In WSNs context, the focus should be on energy savings. In this paper, we propose an adaptive data compression algorithm that tries to satisfy both data quality and energy saving requirements. The paper is organized as follows. Section II describes the main contributions about data compression in sensor networks in order to show the novel aspects of our approach. Section III introduces the main data quality criteria to consider for data compression. Section IV defines the context in which our algorithm is proposed. Details about the algorithm and its performance are provided in Section V and Section VI.

II. Related work

Data compression is a well-established research field, but sensor networks present a context in which new design issues have to be addressed [1]. In fact, the small code and data memories, and the primary focus on energy, call for new approaches [2]. In the traditional research field, a large variety of compression algorithms have been proposed. Most of existing data compression algorithms are, however, not feasible for WSNs owning to their size and complexity. Within the sensor networks community, some data compression studies already exist. For example, there are several contributions (e.g. [3]), where authors address the analysis of high spatial correlation in data from fixed sensors in dense networks. Here, the context is specific and the addressed problems have particular characteristics and criticisms. Our approach aims at handling data from different type of systems (i.e. sparse, dense or mobile) for which data correlation is uncommon. In fact, the proposed algorithm is able to aggregate any data stream characterized by various and unexpected trends. Data compression has been mostly studied to enable in-network processing. In-network processing is the general term used for techniques that process data on a node or group of nodes before forwarding it to the user. The goal of in-network processing of data streams is to select and give priority to

978-1-4244-3304-9/09/$25.00 ©2009 IEEE

reporting the most relevant data gathered [4]. Here, temporal and spatial compression techniques are widely used.

Temporal compression is suitable for all the contexts in which the main goal is to detect the data changes over time. In this scenario, one of the main contribution is a lightweight linear approach [5] [6]. The linear compression algorithm provides a good balance between maximizing compression and minimizing processing complexity for each node. The approach just considers the different measure points at different time instants. Each point is compared with the previous one and it is transmitted only if the measure is significantly different from the previous one. This algorithm is suitable for all the contexts in which phenomena are quite stable and data are characterized by linear trends. In fact, in case of unstable data, the approach would support the communication of all the measured values.

Spatial compression deals with data redundancy in a same physical area. Contributions in this field propose models to discover similar values and to aggregate them by using specific functions [7]. In [8], similar values compose the base signal used to forecast and evaluate the collected data. In spatial compression analysis, the contributions about research on sensors' communication paradigms are extremely relevant (e.g., [9]).

Our approach aims at detecting data changes over time as the linear compression algorithms. The algorithm proposed in Section V is also characterized by a linear complexity and, even if it requires more processing time, it maximizes the compression even when data trend changes frequently. Therefore, it is not so strictly dependent on the phenomenon characterization. Our compression algorithm is based on the concept of time series as [10] [11] [12]. In [10], the authors propose to perform on-line regression analysis over time series on data streams. Autoregressive models built on each sensors are instead used in [11] for forecast time series and approximate the value of sensors readings. Lazaridis and Mehrotra [12] also propose to fit models to time series, but they try to improve system performance, rather than doing regression analysis. We refer to this work, since our model is focused on both quality requirements satisfaction and energy saving. Our model divides the time series in windows and introduces the concept of continuity interval in order to detect permanent data trend changes. Furthermore, our model deals with all types of trends and not only with a limited set as the algorithm proposed by [12]. We also propose a mechanism to change the measure frequency in case of very irregular trends, so increasing the system bandwidth. The adaptation in data stream management is driven by data quality requirements. Several contributions in the literature adopt a similar approach by considering a different set of dimensions. For example, [13] monitors the processing delay to assure data freshness. The total response time is also checked in [14] to optimize the overall QoS performance according to the network condition and work load online. A relevant framework is proposed in [15] in which the precision dimension is used to filter data by using the Kalman filter. In this way they build a flexible system that is able to automatically adapt the reference model to the real-world signal. Authors discard outlier and try to detect the value trend. As for the adaptability to a variety of different types of signals, the contribution in [15] aims at a similar level of flexibility as proposed in this paper. Anyway, differently from [15], in our approach, outliers are important elements of the data stream to consider and store. In addition, we consider other data quality dimensions (i.e., accuracy and timeliness) to improve the efficiency of the algorithm and further reduce the need for transmitting data to the base station.

III. System Quality Requirements

In a wireless sensor network, a set of queries Q is submitted to the base station to get the desired data. The base station injects queries into the wireless sensor network in order to collect sensors' data. In our approach, users can complete the queries with their own quality requirements in terms of *accuracy*, *precision* and *timeliness* dimensions. The behavior of each sensor is influenced by these requirements: each sensor node just collects and sends data in order to satisfy all the requests.

Accuracy is usually defined as the degree of conformity of a measured or computed quantity to its actual (true) value. Accuracy is related to *precision* that is the degree to which further measurement or calculations show the same or similar results [16]. The impact of these two dimensions on data stream management is discussed in Section IV. *Timeliness* is defined as the property of information to arrive early or at the right time. Timeliness is usually measured as a function of two elementary variables: currency and volatility [17][18]:

$$Timeliness = max(1 - Currency/Volatility; 0)^s$$

where the exponent s is a parameter necessary to control the sensitivity of timeliness to the currency-volatility ratio. In the analyzed context, *currency* can be defined as the interval from the time the value was sampled to the time instant at which data are sent to the base station. *Volatility* is a static information that indicates the amount of time units (e.g., seconds) during which data remain valid. Volatility is usually associated with the type of phenomena that the system has to monitor and depends on the change frequency. Timeliness constraints are one of the main drivers for data processing and transmission. When users submit queries, they have to define their quality requirements. As an example, the PERLA language [19] allows a conditional execution of operations on the basis of quality parameters.

IV. System and data structure

The design of a sensor network is strongly driven by its particular application. WSNs can be used for data collection purposes in situations such as environmental monitoring, habitat monitoring surveillance, equipment diagnostics, disaster management, and emergency response [20]. In our context, we analyze systems for environmental monitoring, specifically we consider phenomena in which the change frequency is low.

The input data stream can be seen as a continuous flow of real-time data tuples of the form <*sensor-id, timestamp,* *value*> coming to the sensor's input buffer. As in many real-time systems, we can suppose that the Input Buffer (IB) is actually split into two separate storage areas (i.e., IB_1 and IB_2). Data are fed to IB_1 until it is full or timeliness requirements force data processing, then input is switched to IB_2 while data in IB_1 are transferred to the compression engine and then to the output buffer as in Figure 1. Then the switching process is repeated and data are processed from IB_2.

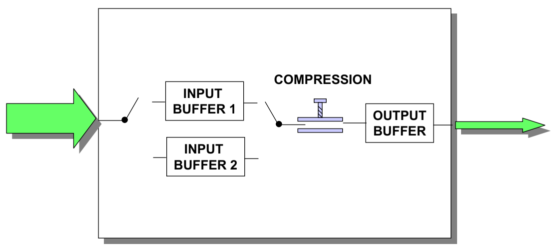

Figure 1. System architecture

In this way, the potentially infinite data stream is reduced, with a windowing approach, to a sequence of finite time-ordered data sets on each of which the compression algorithm can work. In our approach, the main window can be further partitioned into smaller sub-windows (see Figure 2) whose values are considered so *similar* that they can be aggregated by computing their average ("X" values).

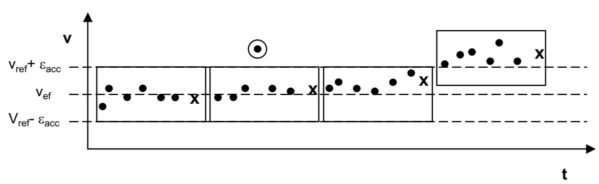

Figure 2. Windowing on data stream

Only the average values and the outliers will be transferred to the base station. The size of the sub-windows depends on the similarity of the values involved in the data stream and on timeliness constraints. The similarity concept is defined by considering the *accuracy* and *precision* dimensions. Accuracy can be defined as the windows height; in particular, the window height controls the accuracy of the measure and the robustness in finding *outlier values*, i.e. those values which depart from the normal trend and could mean either measurement errors or a change in the measured system behavior. The two cases can be automatically distinguished by considering the precision value. Indeed, if the values are not accurate, but precise, it means that values are not similar to the reference value, but they are characterized by a small standard deviation. In this case a change in the measured system has occurred. A measurement error is an occasional event and the values are still judged as accurate and precise. It is also possible to consider the case in which data values are not accurate and precise and this occurs when the change in the measured system is slow or not regular. In Section V, the different cases are formally described.

The sub-window width also depends on timeliness requirements and controls the compression amount: the larger the number of points in the window, the larger the compression we get, but also the larger the transmission delay of time sensitive data. A window with a single point does not compress data. Notice that a reduction of the window's width is tantamount to increase the measure frequency (system bandwidth) in order to catch sudden changes.

V. The model

In this part, we focus on the interaction between a single data producer (sensor) and a data archiver (database located in the base station). The proposed algorithm aims at modelling the sensor's behavior; we do not focus on data processing and mining performed by the base station.

A. Preliminaries

Each sensor has a sampling period that defines the time instant t_i in which data are acquired. Considering these time instants we define a value series V $=< v[1], v[2], ...v[n] >$ as a collection of values observed in subsequent *n* time instants. The maximum number of measure points *N* coincides with the cardinality of data in the input buffer. Sub-windows are characterized by their width *W* and height *H*. The former is the number of points in a sub-windows, it expresses the compression factor and depends on timeliness requirements. The latter is the biggest difference between two measure points in a window and controls the accuracy of the measure and the robustness in finding outlier values. Note that an outlier is a value which departs from the normal trend and could mean either measurement errors (the circled point in Figure 2) or a change in the measured windows (the last two sub-windows in Figure 2).

The quality oriented approximation starts from the definition of the requirements about accuracy and precision. Ideally, a measure device is both accurate and precise, with measurements all close to and tightly clustered around the reference value. In continuous value monitoring, accuracy and precision can be used as the features that reveal errors or changes in the monitored process. Precision is often characterized in terms of the standard deviation of the measured values. The smaller the standard deviation, the higher the precision. We can instead define accuracy as the error expressed by the difference between the mean of the measurements and a reference value.

As stated in Section IV, it is important to evaluate the combined effect of accuracy and precision. It is possible to distinguish different situations along the values of the two dimensions (see Figures 3):

a) *Expected trend*: the trend is regular since values are precise and accurate;
b) *Slow change*: the trend is characterized by an unexpected, but lasting variation. Values are still precise, but not accurate;
c) *Oscillatory/bursty trend*: the trend is characterized by unexpected and discontinuous variations. Values are not precise, but they can be both accurate or inaccurate.

Note that any data stream can be described as the combination of the trends described above. The sensor has a finite energy

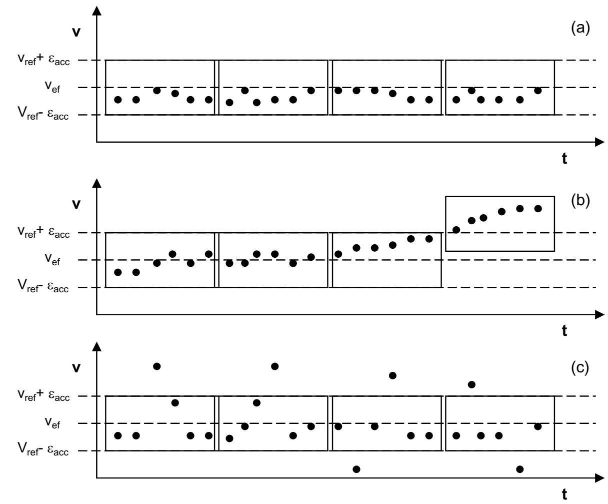

Figure 3. a) Expected trend, b) Slow Change, and c) Oscillatory/bursty trend

supply. This is depleted during normal sensing operation at some rate. Additional energy drain is caused when using any of the sensor equipment, including (i) powering its memory, (ii) using its CPU, (iii) sending/receiving data. The rates of energy consumption for these operations are sensor-specific. Communication is often the major cause of energy drain in sensors and hence, in the interest of extending the sensor's life, communication must be limited. Hence, each sensor is characterized by:

- e_t: energy consumption for the transmission of one byte;
- e_e: energy consumption for processing one instruction;
- E_t: energy consumption for data transmission to the base station;
- E_e: energy consumption for processing analysis and aggregation algorithms;
- E_{tot}: total energy consumption of the sensor node, calculated as $E_t + E_e$.

The model presented in the following aims at minimizing the total energy consumption by considering that $e_t \gg e_e$. In fact, the algorithm analyzes the values in the data stream in order to define if it is possible to communicate to the base station only the aggregate values and the outliers. By considering Z aggregate values and J outliers, the algorithm is considered efficient if the output is composed by (Z+J) values instead of N where (Z+J)$\ll$ N.

B. The algorithm

This section describes in details our approach by showing its feasibility and adaptivity.

1) Setting the input parameters: Quality requirements are the main input to provide for the data processing. As stated in Section V quality requirements concern accuracy, precision, and timeliness dimensions. Formally, we define accuracy as the difference between the mean of the measurements v_n and the reference value v_{ref}, the bias [16]. The acceptable measurement error can be defined as ε_{acc} and the measure is considered accurate if:

$$|v_n - v_{ref}| < \varepsilon_{acc}$$

Note that the ε_{acc} specification defines the height of the sub-windows since $H = 2 \cdot \varepsilon_{acc}$.

The precision can be defined as the reciprocal of variance. A measure is considered precise if:

$$1/n \cdot \sum_{n=1}^{N} (v_n - \mu)^2 < \varepsilon_{prec}$$

where μ is the average of the values v_n.

Another required input is *N*, i.e. the maximum number of points to be considered in a sub-window. Furthermore, it is necessary to specify the number of observations *C* needed to evaluate if a change in a trend is permanent or transient.

2) Algorithm body: Figure 4 describes inputs and outputs of the algorithm.

	Name	**Symbol**
INPUT	Time Series	V= $\langle v_1, v_2, ...v_n \rangle$
	Expected value	v_{ref}
	Accuracy tolerance	ε_{acc}
	Precision tolerance	ε_{prec}
	Window Width	N
	Continuity interval	C
OUTPUT	Aggregate values	T= $\langle t_1, t_2, ...t_z \rangle$
	Outliers	O= $\langle o_1, o_2, ...o_j \rangle$

Figure 4. Input and output data

The algorithm body works as described in the following. The algorithm stores in a vector all the incoming measure points v_n. The number of points depends on the window width.

(1) Indata[n] = v_n

Check accuracy and precision and evaluate the different cases. The algorithm analyzes the different data streams trends described in Figure 3.

(2) SWITCH(accuracy, precision)

CASE 1: Data follow the expected trend. If we have analyzed all the measure points in the buffer, the average of the acceptable values is calculated. The acceptable values are defined as all the stored values except for outliers. If the number of outliers is more than half of analyzed values then all the values will be transmitted without any further analysis.

(3) CASE ($< \varepsilon_{acc}$, $< \varepsilon_{prec}$)

(4) IF number of analyzed values=N AND Number of outliers $< N/2$ THEN t[z]=AVG(Acceptable values); Increment z

(5) ELSE IF Number of outliers $> N/2$

(6) THEN T= $\langle t_1, t_2, ...t_z \rangle$= V= $\langle v_1, v_2, ...v_n \rangle$

(7) ELSE Analyze new value and GO TO (1)

CASE 2: Data undergo a slow change. In this case when the algorithm detects an outlier, it controls if it is associated with a permanent or transient data trend change. In the former case, it calculates the average of the values stored before the exception and recalculates the expected value v_{ref} along the last inaccurate values. In the latter case, inaccurate values are transmitted to the base station as outliers.

(8) CASE ($> \varepsilon_{acc}$, $< \varepsilon_{prec}$)
(9) Variable initializations: the number of unexpected values, the time instant in which the exception occurs (T_e)
(10) O[j]=Indata[n]
(11) Indata[n]= v_{n+1}
(12) WHILE accuracy $> \varepsilon_{acc}$ AND precision $< \varepsilon_{prec}$
(13) INCREMENT the number of unexpected values
(14) O[j]=Indata[n]
(15) IF number of analyzed values=N AND number of subsequent unexpected values = C
(16) THEN t[z]=AVG(Acceptable values arrived before T_e); Increment z
(17) t[z]=AVG(Acceptable values arrived after T_e); Increment z
(18) v_{ref}=AVG(Acceptable values arrived after T_e)
(19) Delete the last C O[j] and GO TO (1)
(20) ELSE IF number of analyzed values=N AND number of subsequent unexpected values<C
(21) THEN t[z]=AVG(Acceptable values); Increment z
(22) ELSE IF number of subsequent unexpected values = C
(23) THEN t[z]=AVG(Acceptable values arrived before T_e); Increment z
(24) v_{ref}=AVG(Acceptable values arrived after T_e)
(25) Delete the last C O[j] and GO TO (1)
(26) else Indata[n]=v_{n+1} and GO TO (12)
(27) GO TO (2)

CASE 3: Data are characterized by an oscillatory trend or bursts. The analyzed value is classified as an outlier.

(28) CASE ($< \varepsilon_{acc}$, $> \varepsilon_{prec}$) OR ($> \varepsilon_{acc}$, $> \varepsilon_{prec}$)
(29) O[j]=Indata[n]
(30) IF number of analyzed values=N AND Number of outliers $< N/2$ THEN t[z]=AVG(Acceptable values); Increment z
(31) ELSE IF Number of outliers $> N/2$
(32) THEN T= $\langle t_1, t_2, ...t_z \rangle$= V= $\langle v_1, v_2, ...v_n \rangle$
(33) ELSE Analyze new value and GO TO (1)

3) Energy Consumption: The proposed algorithm can be considered efficient if:

$$e_t \cdot N > E_c + e_t \cdot Z + e_t \cdot J$$

This condition is not valid in case the number of outliers is greater than $N/2$. In fact, in this case the energy consumed will be the sum of $e_t \cdot N$ and E_c. This situation occurs when the data trend is irregular; in such a case, the measurement frequency of the sensor should be increased in order to identify the small data variations. In fact, on the basis of the received data, the base station must reconsider the context conditions and adapt the algorithm by identifying the new most suitable measurement frequency for the considered sensor.

Note that it is also possible to influence the number of outliers and the accuracy of compression by defining a suitable C value. In fact, a high C value increases the probability of classifying values as outliers instead of trend changes. Thus, the probability to send all the N values increases as well.

VI. Experimental results

This section evaluates the performances of the proposed algorithm in terms of functional features and energy consumption properties.

A. Compression experiments

The algorithm has been implemented in C language and tested in order to check its correctness and efficiency. Three different data set are considered. Each data set corresponds to a time series representing one of the trend described in Section V: (i) Expected trend, (ii)Slow change, and (iii) Oscillatory/Bursty trend. Furthermore, different window sizes have been considered to show their influence on the precision of the results. In all the experiments, the values shown in Figure 5 have been used as input data. In Figure 6, expected trend

Name	Value
Number of values	35
v_{ref}	50
ε_{acc}	5
ε_{prec}	5
N	10,15,20,25,30,35
C	5

Figure 5. Experimental Input data

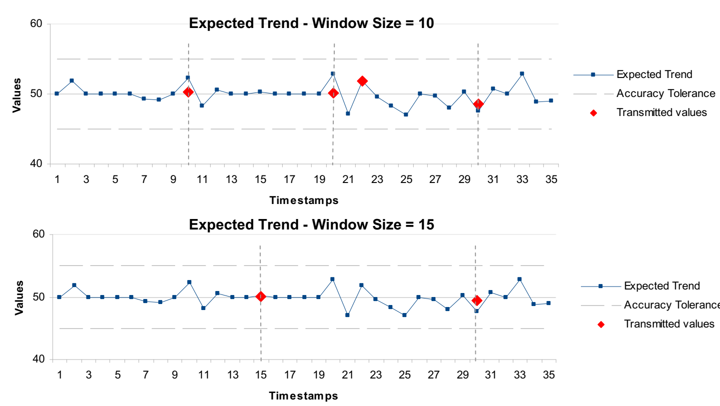

Figure 6. Transmitted values in case of Expected Trend

is shown. Note that only aggregates values are transmitted. In case of window size equal to ten values, the algorithm found an outlier due to the fact that precision requirements were not satisfied. In Figure 7, slow change trend is shown. The algorithm sends all the values that correspond to a change from the expected trend. Note that also the reference value changes along the analyzed data set. In Figure 8, oscillatory trend is shown. The algorithm sends all the values in a window when the majority are outliers.

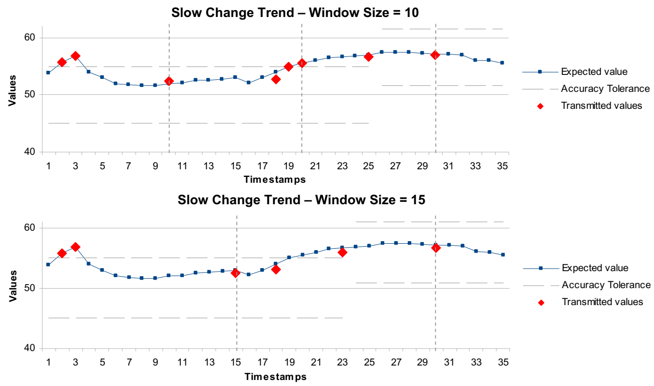

Figure 7. Transmitted values in case of Slow Change Trend

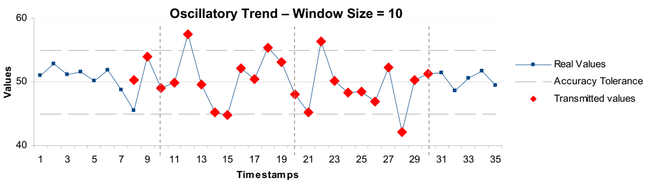

Figure 8. Transmitted values in case of Oscillatory/Bursty Trend

B. Energy savings

We calculate the energy consumed by the sensor processor by using Avrora that is a set of simulation and analysis tools for programs written for the AVR microcontroller produced by Atmel and the Mica2 sensor nodes. We consider Mica2 sensors that are powered by two AA alkaline batteries and are built around an Atmel Atmega 128L microcontroller circuit and the CC1000 integrated radio circuit. The processing energy has been added to the transmission energy in order to demonstrate that the total energy consumed is lower than the one consumed by transmitting all the values. As to transmission energy, data provided in [21] have been used. In details transmission energy per bit is used, which sets the required transmission energy at 4.28 μjoule/bit.

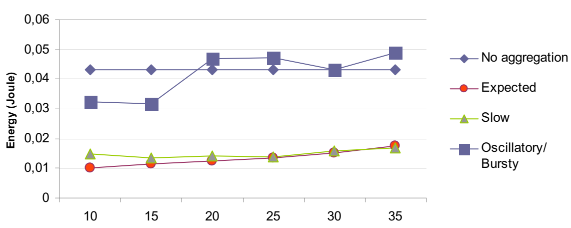

Figure 9. Energy comparison

Figure 9 shows that the algorithm allows users to save energy. In case of oscillatory trends, the algorithm fails when data trend is irregular. In this case, the measurement frequency of the sensor has to be changed in order to catch the data variations.

VII. Conclusions

Economic usage of energy and computational power in WSNs are significantly bigger issues than in normal networks. Data management must take care of energy drain by optimizing transmission operations among sensors, which are the most energy consuming, while keeping their quality as high as possible. While first encouraging results have been presented in this paper, thorough analysis of the algorithm's performances with real data streams on physical devices will be performed. Further work will also improve the algorithm by focusing on the definition of an optimization model for the maximization of energy savings and the automatic definition of the most suitable continuity interval value.

Acknowledgments

This work has been partially funded by the MIUR-FIRB ART_DECO and by the SMScom projects at Politecnico di Milano.

References

[1] N. Kimura and S. Latifi, "A survey on data compression in wireless sensor networks," in *ITCC*, 2005, pp. 8–13.
[2] L. Golab and M. T. Özsu, "Issues in data stream management," *SIGMOD Record*, vol. 32, no. 2, pp. 5–14, 2003.
[3] J. Chou, D. Petrovic, and K. Ramchandran, "A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks," in *INFOCOM*, 2003.
[4] J. Gama and M. M. Gaber, *Learning from Data Streams*. Springer, 2007.
[5] R. Cardell-Oliver, "Rope: a reactive, opportunistic protocol for environment monitoring sensor networks," in *EmNetS-II*, 2005.
[6] T. Schoellhammer, E. Osterweil, B. Greenstein, M. Wimbrow, and D. Estrin, "Lightweight temporal compression of microclimate datasets," in *LCN*, 2004, pp. 516–524.
[7] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: A tiny aggregation service for ad-hoc sensor networks," in *OSDI*, 2002.
[8] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Compressing historical information in sensor networks," in *SIGMOD Conference*, 2004, pp. 527–538.
[9] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *MobiCom*, 2000, pp. 56–67.
[10] Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang, "Multi-dimensional regression analysis of time-series data streams," in *VLDB*, 2002, pp. 323–334.
[11] D. Tulone and S. Madden, "Paq: Time series forecasting for approximate query answering in sensor networks," in *EWSN*, 2006, pp. 21–37.
[12] I. Lazaridis and S. Mehrotra, "Capturing sensor-generated time series with quality guarantees," in *ICDE*, 2003, pp. 429–439.
[13] Y.-C. Tu, M. Hefeeda, Y. Xia, S. Prabhakar, and S. Liu, "Control-based quality adaptation in data stream management systems," in *DEXA*, 2005, pp. 746–755.
[14] H. Hu, C.-H. Jiang, K.-Y. Cai, and W. E. Wong, "A control-theoretic approach to qos adaptation in data stream management systems design," in *COMPSAC (2)*, 2007, pp. 237–248.
[15] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using kalman filters," in *SIGMOD Conference*, 2004, pp. 11–22.
[16] *ISO/IEC Guide 99-12:2007 International Vocabulary of Metrology Basic and General Concepts and Associated Terms*, ISO.
[17] H. P. D.P. Ballou, R.Y. Wang and G. Tayi, "Modelling information manufacturing systems to determine information product quality," *Management Science*, vol. 44, no. 4, 1998.
[18] R. S. M. Bovee and B. Mak, "A conceptual framework and belief-function approach to assessing overall information quality," in *ICIQ*, 2001.
[19] F. A. Schreiber, R. Camplani, M. Fortunato, M. Marelli, and F. Pacifici, "Perla: A data language for pervasive systems," in *PerCom*, 2008, pp. 282–287.
[20] D. E. Culler, D. Estrin, and M. B. Srivastava, "Guest editors' introduction: Overview of sensor networks," *IEEE Computer*, vol. 37, no. 8, pp. 41–49, 2004.
[21] M. Calle and J. Kabara, "Mac protocols for gsp in wireless sensor networks," *JNW*, vol. 3, no. 6, pp. 29–35, 2008.