Data Cleansing and Selective Query Processing in Sensor Networks

Maria Drougka, Theodore Tsiligiridis
Division of Informatics, Mathematics and Statistics,
Department of General Science, Agricultural University of Athens,
Athens, Greece, EU
{*mdrougka, tsili*}*@aua.gr*

***Abstract*—Wireless sensor networks have been widely used in numerous monitoring applications and real life phenomena. Due to the low quality of sensors and random effects of the environment, the collected sensor data is subject to several sources of errors. Such errors may seriously impact the answer to any query posed to the sensors and therefore, it is very critical to clean the sensor data before using them to answer queries or conduct data analysis. Well known data cleansing approaches, are used to meet the requirements of both energy efficiency and quick response time in many sensor related applications. Several energy saving methods based on clustering sensors, so that sensors communicate information only to cluster-heads and then the cluster-heads communicate the aggregated information to the processing center are reviewed, and discussed. We focus on the distribution of sink operation among sensor nodes and specify the criteria of selecting effective cluster-heads that enable both balanced and low energy consumption in data query and collection. As a result of the effective selection of cluster-heads, both balanced and decreased energy consumption are superior compared with the conventional retrieval model, which is not using cluster-heads, because their use provide maximum data aggregation among various sensors, and it alleviates the heavy energy consumption near the sink.**

***Keywords*-Data cleansing, selective query, Wireless Sensor Network.**

I. Introduction

Wireless Sensor Networks (WSN) have become an important source of data with numerous applications in monitoring various real-life phenomena as well as agro-environmental and industrial applications. Data delivered by sensors is typically noisy and unreliable. The task of improving, correcting and filtering the incoming data is usually referred as *data cleansing*. Such errors may seriously impact the answer to any query posed to the sensors, since they may yield imprecise or even incorrect and misleading answers, which can be very significant if they result in immediate critical decisions or activation of actuators.

A. Dimensionality of Data Cleansing

Data cleansing has different dimensions, which are related to the different kinds of noise.

- *Noise Reduction or Smoothing:* Stochastic noise or noise caused by interferences can often be treated by some kind of smoothing (like a moving average) or other noise reduction methods. The idea is to extract the relevant portion of the incoming signal. The incoming signal is modified but not removed. Thus the risk of loosing relevant information by this kind of cleansing is very low.
- *Filtering:* Faulty data due to sensor failures or human errors can often be recognized by cross-checking information from different sources. This is often based on physical models. In certain places for example one could use temperature and smoke sensors to detect fire. Cross-checking the signals from spatially close sensors of both kinds allows to detect unlikely or impossible measurements of a single sensor and could help to avoid false-alarms. However filtering has to be used carefully as it might be vulnerable for manipulations.
- *Generation:* Data cleansing may also try to fill gaps in the data due to missing sensors or transmission errors. This task is likely to employ physical models. Filling up missing data can be very useful as it allows to draw an overall picture of the state of an infrastructure. Such data should be marked as generated, virtual or simulated since it is not as reliable as real measurements.

B. Methods for Data Cleansing

In order to remove noisy sensor data or at least reduce the effect that brought about by noises is a key issue to answer queries or detect events accurately. Statistical and probabilistic modelling techniques have been used to solve the issues we discussed earlier. Modelling usually involves two phases: training and testing. In training, the parameters of the characteristic function representing the data are learned. Sometimes held-out data is used for validation to further improve the accuracy of the training process by preventing over-fitting. In the next phase, predictions are made about the testing data. Training is frequently done off-line while testing can be done either off-line or on-line.

Sensor data is temporal and spatial in nature. In general, a reading is usually of the format <*sensor-id, location, time, value*>. If the sensors are static, then the location field is usually omitted. Individual observations are assumed to be

independent. Traditional data cleansing techniques cannot be applied to sensor data as they do not take into account the strong spatial and temporal correlations typically present in sensor data. Nevertheless, well known data modelling methods like Kalman filters and regression have shown good results in capturing spatio-temporal correlations. The Kalman filter is an efficient recursive filter, which estimates the state of a dynamic system from a series of incomplete and noisy measurements. Regression usually involves fitting the best curve for a given set of points. In the case of time-varying and spatial data the use of regression is mainly to find the best curve approximating the readings. This curve can be used not only to find missing or unknown data but also to reduce noise.

So far, interesting prior work has been done on data cleansing in the context of sensor network data. Many toolkits for cleansing noisy and incomplete sensor data are available, supporting interpolation and extrapolation functionalities. In addition, they provide data analysis tools like visualization and a step utility to examine the actual training process. Comparative studies of two methods shows promising results for the Kalman filter for most of the physical attributes modelled.

C. Cleansing and Query processing

If cleansing is performed at the sensors, there would be significant communication cost in sending the parameters of the model to the individual sensors. Furthermore, there is a storage cost associated with storing the parameters. In addition to communication and storage costs, performing the actual cleansing at the sensors would incur a processing cost on the resource-constrained sensors. These problems do not arise if cleansing is done at the sink, given the typical processing power and storage capacity of sinks. Moreover, there would be huge savings by not having to communicate the model parameters to each individual sensor. Given that the lifetime of the sensors is heavily dependent on the amount of communication that they do, communication savings is very important. Having the data and the model at the sink is advantageous when it comes to query processing, as answers to user queries can be easily computed.

We define a monitoring query as a continuous data collection task that requests sensed values from nodes fulfilling selection criteria based on certain physical conditions. For instance, queries that monitor object movements in a field would report the sensed values only from the nodes that have recently sensed these movements. The following are key aspects of monitoring queries:

- *Monitoring queries are selective:* Typically, a WSN can cover an area much larger than the area of interest at any given point in time. For instance, in the monitoring example presented above, although nodes are present all over the field, the object movements may only be found within a limited area. We argue that for energy efficient optimization of such queries, the data collection task should be selectivity-aware.
- *Monitoring queries are continuous:* A monitoring task, by design, is expected to query readings from sensor nodes over an extended period of time. The mainstream WSN database systems have realized the need for continuous queries and provide SQL clauses to define such queries.
- *Monitoring queries select spatially correlated nodes:* Physical phenomena are characterized by their spatial correlation; hence, when monitoring a physical phenomenon, sensor nodes at proximal locations tend to have similar values. Therefore, this spatial correlation, coupled with the notion of selectivity, results in clustered participation. For instance, if a node is selected by a query based on its sensed temperature value, there is a high probability that neighbouring nodes will also be selected by the same query.

Cleansing and query processing can be performed either at the individual sensors or at the sink. Performing the cleansing at the sensor level and query processing at the sink has no clear advantages. This is because communicating a single noisy reading to the sink and performing the cleansing work there incurs less communication cost than communicating all the model parameters over to the sensor itself. The latter, as we have mentioned earlier, imposes unnecessary processing and storage overhead on the sensors.

The remainder of the paper is structured as follows: Section II summarises the background information related with the existing cleansing and querying mechanisms, Section III analyses, in terms of minimization of the total energy spent in the system, two schemes proposed for data collection and querying process, and finally, Section IV concludes the present work with some discussion on the results and a future work.

II. Material and Methods

A. Cleansing Mechanism

1) Weighted moving average algorithm: A well known approach to remove noise in random samples and compute the monitoring values is to use the moving average [1], [2]. Note that moving average in sensor networks has two dimensions. Sensor data are averaged temporally within one sensor, and also spatially among neighbouring sensors. For example, at any time t, the algorithm first averages a sequence of samples $x_{i,t-k+1}, \ldots, x_{i,t}$ at each sensor i, and gets $\bar{x}_i = (x_{i,t-k+1} + \ldots + \bar{x}_{i,t})/k$. We then average values of neighbouring sensors, $\bar{\bar{x}}_i = \sum_{j \in R(i)} \bar{x}_j / |R(i)|$, where $R(i)$ is a set of neighbouring sensors of sensor i.

The above approach is not suitable for sensor network applications, mainly because there is a trade-off between energy efficiency and query response time. For example, to improve the efficiency sampling rates should be low, namely,

the interval between two consecutive samples should be long. Thus, any change takes a long time to be reflected in the moving average. On the other hand, if the sampling rate is high, the response to a change is short and therefore more samples need to be taken. However, in practice, since sampling is one of the costly operators for sensors, the energy efficiency of high sampling rate is lowered. To address these two aspects together a weighted moving average algorithm has been proposed [3], that collects confident data from sensors and computes the weighted moving average.

In particular, the *temporal moving average* is defined as:

$$\bar{x}_i^t = (w_{i,t-k+1}x_{i,t-k+1} + \ldots + w_{i,t}x_{i,t})/h \qquad (1)$$

where $w_{i,t}$ is the weight of value $x_{i,t}$ at sensor i, which is related to the confidence of this value and $h = w_{i,t-k+1} + \ldots + w_{i,t}$ is the accumulated temporal weight. In addition, the *spatial moving average* of the data coming from spatially correlated sensors is defined as:

$$\bar{x}_i^s = \frac{1}{m} \sum_{j \in R(i)} b_{j,t} w_{j,t} x_{j,t} \qquad (2)$$

where $m = \sum_{j \in R(i)} b_{j,t} w_{j,t}$ is the accumulated weights and weight $b_{j,t}$ is a digit value according to whether or not a neighbouring sensor j reports the confident value $x_{j,t}$ to the sink. We can greatly achieve low sampling cost when $x_{i,t}$ is a smooth and predictable value. Data cleaning is performed in two places. On the sensor side, multiple sampling is used to remove random noises in data, whereas on the sink side, the weighted moving average is used in both dimensions to further smooth the data.

From equation (1) the temporal part is a weighted version of the normal moving average $\bar{x}_i^t$ provided that $w_{i,t} = \hat{w}_{i,t} \forall i, t$ and k is a given window size. The spatial part includes value $x_{j,t}$ from sensors j, where $x_{j,t}$ is in the error range $[x_{j,t} - e, x_{j,t} + e]$ of $x_{i,t}$, and it has a high confidence with its weight $w_{j,t} > 1$. That means $x_{j,t}$ is in $x_{i,t}$'s error range and has high confidence to improve the moving average at $x_{i,t}$. In a similar way the spatial part of moving average is provided by equation (2). Finally the weighted moving average is the combination of the temporal and spatial parts: $\bar{x}_{i,t} = (h\bar{x}_i^t + m\bar{x}_i^s)/(h + m)$.

2) k-Means Algorithm: The latest sensor network techniques enable a sensor to sense multiple measures simultaneously. Therefore, multidimensional data analysis such as clustering is needed for analyzing sensor network data. For example, for temperature sensor network the dimension is one, for a sensor network measuring both temperature and humidity the dimension is two, whereas for a sensor network measuring temperature, humidity, and density at the same time the dimension is three, etc. k-means algorithm proposed in [4] and [5], is an old, simple, and well known method for analysing multidimensional data by separating data into different clusters. It converges very fast when the dimension of data is small, such as in the cases of environmental problems.

In mathematical terms the problem is defined as follows. We consider a set of points U, where U is assumed to be an r-dimensional space. At a time instant i, the value of a point $u \in U$ is $u^i = (u_1, \ldots, u_r)$. In other words, a point in U can be regarded as a moving object in an r-dimensional space. We are interested in k-means clustering of the current values of the points at each time instant. At an instant i, let $c_1, \ldots, c_k$ be k points, which may or may not be in U. The points in U can be partitioned into k exclusive subsets $U_1, \ldots, U_k$ according to their values at instant i: a point $u \in U$ is assigned to cluster U_i if:

$$dist(u^i, c_i) = \min_{1 \le j \le k} \{dist(u^j, c_j)\} \qquad (3)$$

where $dist()$ is the distance function in question. Note that the points $c_1, \ldots, c_k$ are the k-means of U if:

$$\min \left\{ \sum_{i=1}^{k} \sum_{u \in U_i} dist(u^i, c_i) \right\} \qquad (4)$$

To lower down the communication cost a hierarchical clustering structure is proposed in [6] and [7]. A set of sensors are grouped together, and one of them becomes a Cluster-Head (CH). In data collection, each sensor in the cluster sends its data to the CH, and the CH reports the aggregated data to the sink. A distributed randomized algorithm was proposed to cluster the sensors. Each sensor takes a probability to become a CH, and broadcasts itself to other sensors within certain hops. The sensors tha t are not CHs join the closest CH. The optimal parameters of the clustering, which minimize the communication cost are also derived. However, as time goes by, the status of each sensor may change, and thus the so-built hierarchical structure may not always be optimal. For example, some sensors may use more energy to collect data, so they are dying faster than the others. If we use such sensors as CHs, the lifetime of the whole cluster decreases. The problem has been studied in [8], where it periodically recomputes the CHs based on the residual energy of each sensor and its relationship to other sensors.

To maximize the lifetime of the sensor network a hierarchical model is used that utilizes data aggregation and in-network processing at two-levels of the network hierarchy. First, a set of sensor nodes called Local CHs (LCHs) are elected to form a fixed virtual routing architecture on, which the first level of aggregation and routing is performed. Then, the problem is that of finding an optimal subset of LCHs, called Master CHs (MCHs), which are selected to perform the second level of aggregation under the objective to maximize the network lifetime. Clearly, the problem of optimal selection of MCHs is NP-complete since it is equivalent to the p-median problem in graph theory, which has been shown to be NP-complete [9].

To implement the k-means algorithm one needs to initialize mean values with k, say, random MCHs from the set of LCHs. To initialize k MCHs we choose k LCHs at random from the set of sensors S, $|S| = s \in \mathbf{N}$ while making sure that the pairwise distance (number of hops) between these k MCHs is large enough. One way to do that is to choose m LCHs at random from S (where $m >> k$ but $m < s$) and then perform k-means clustering on those m LCHs. To initialize this sub-problem, we arbitrarily assign LCHs to different k MCHs(classes). The algorithm simply iterates until a termination condition is met. In every iteration, each LCH in S is assigned to a chosen MCHs such that the distance from LCHs to sink through that selected MCHs is minimized. Then, for each class, we recalculate the means of the class based on the local nodes that belong to that class. Theoretically, k-means should terminate when no more LCHs are changing classes; however, in practice, this may require a large number of iterations.

3) Weighted data cleansing: The above approach introduces a periodic multilevel data cleansing algorithm aiming to optimize the volume of data transmitted thus saving energy consumption and reducing bandwidth on the network level. It is based on a tree network where sensed data needs to be aggregated on the way to their final destination. In particular, a frequency filtering technique is applied, which exploits the ordering of measurements according to their frequencies, by means of the number of occurrences of this measure in the set. Since sensor nodes are deployed randomly, it is most likely that neighbouring nodes generate similar sets of data.

Data aggregation works in two phases, the first one at the LCHs, where each node compacts its measurements set according to a link function. The objective is to identify similarities between neighbouring sensor nodes, and integrate their sensed data into one record while preserving information integrity. This first level of cleansing process is called *in-sensor process periodic cleansing*. A second cleansing process is applied on the level of the MCH itself, where the frequency filtering technique will be applied. The cleaned data is finally sent from the MCHs to the sink.

In periodic sensor networks, at each period T each node sends its aggregated data set to its proper LCH, which subsequently aggregates all data sets coming from different sensor nodes and sends them to the sink. We consider that each sensor node i at each slot takes a new measurement y_t^i. Then node i forms a new set of sensed measurements M_i with period T, and sends it to the aggregator. Note that, a sensor node can take different kind of measures (e.g., temperature, humidity, light, etc), making of y_t^i a vector instead of a scalar. For the sake of simplicity, in the rest of the paper we shall consider that $y_t^i \in \mathbf{R}$. It is likely that a sensor node takes the same (or very similar) measurements several times especially when t is too short. In this phase of aggregation, we are interested in identifying duplicate data measurements in order to reduce the size of the set M_i. Therefore, to identify the similarity between two measures, we define the link function between two measures as:

$$link(y_{t_j}^i, y_{t_k}^i) = \begin{cases} 1 & : \quad \text{if } \|y_{t_j}^i - y_{t_k}^i\| \leq \delta \\ 0 & : \quad \text{otherwise} \end{cases} \tag{5}$$

where δ is a threshold determined by the application. Furthermore, two measures are similar if and only if their link function is equal to 1. The frequency of a measurement y_t^i is defined as the number of the subsequent occurrence of the same or similar (according to the link function) measurements in the same set. It is represented by:

$$f(y_t^i) = \sum_{j=t_i+1}^{T} link(y_{t_j}^i - y_{t_k}^i)$$

4) Greedy Algorithm: In the virtual graph of the set of sensors S, $|S| = s \in \mathbf{N}$, LCHs are numbered sequentially from 1 (left-upper corner) to s starting from 1 and then from left to right proceeding row by row. The greedy algorithm starts with the first node in S and proceed sequentially through the whole topology in a left-to-right and top down fashion [10]. We assume that each LCH has global information about network topology (shortest path from each LCH - LCH and from LCH - sink can be obtained by running all-pairs shortest path algorithm) and this information is broadcasted before the greedy algorithm is executed. To construct the MCHs graph, the shortest path is first established for the first LCH source node acting as the first MCH to the sink. Each subsequent LCH source is incrementally connected to the MCHs graph either as a MCH itself or by selecting a MCH from the set of nodes that are already been allocated as MCHs. In the latter, a LCH selects the MCH that results in the least power consumption to reach the sink. Then, LCH sends the MCH its group number. A MCH holds a registry for its constituent LCH and for those LCHs that exist in multiple groups to distinguish data coming common LCHs. The process is iterated until all LCHs are covered by MCHs and until there is no change in the value of the objective function, which is to obtain the least total power consumption to reach the sink.

B. Query Resolution Mechanism

When the sink receives a query message, it resolves the name of the query to the corresponding sensor IDs, according to the resolution table. After a query′s name is translated into an ID group corresponding to sensors, the sink calculates the query area by deriving a rectangle, in which all corresponding sensor nodes reside. A rectangle area is calculated based on the locations (x_i, y_i) of each sensor $i = 1, 2, \ldots, s$ that is in the ID list obtained from the query resolution: $R_{xy} = R_x \times R_y = [min\{x_i\}, min\{y_i\}, max\{x_i\}, max\{y_i\}] \quad \forall i = 1, \ldots, s$ with $s = |S|$ be the number of all sensors in the network. It is assumed that all sensors can reach the sink, using multi-hop

communication. In addition, whenever a sensor $j = 1, \ldots, s$ is activated it emits a constant energy signal E_j in the surrounding environment. The measured signal is inversely proportional to the distance from the activated sensor raised to some power $\gamma \in R^+$, which depends on the environment. As result, the measurement of a CH $h \in H, H \subseteq S$ is given by the equation:

$$z_h = min \left\{ E^{ch}, \sum_{j=1}^{s} \frac{E_j}{r_{hj}^{\gamma}} \right\} + w_h \tag{6}$$

where E^{ch} is the maximum energy, which can be recorded by a CH, r_{hj} is the radial distance of CH h from the sensor $j \in S$, $r_{hj} = \sqrt{(x_h^{ch} - x_j)^2 + (y_h^{ch} - y_j)^2}$ and w_h is additive Gaussian noise with zero mean and variance σ_h^2. Note that the neighbourhood of a CH h is defined as the set of all sensors that are located at a distance less than or equal to r_c. Therefore the neighbourhood of a CH $h \in H$ is the set of all sensors in S that are in the disk centred at $\vec{x}_h$ with radius r_c, or,

$$CH_{r_c}(h) = \{j : \|\vec{x}_h - \vec{x}_j\| \leq r_c, \forall j \in S, j \neq h\}, \ \forall h \in H$$

Thus, in case r_c is the communication range of the sensor, then the set $CH_{r_c}(h)$ defines all sensors that are one hop away from the CH h.

Generally, sensing data is collected at appropriate CHs, at which is aggregated and forwarded to the sink. There are two types of CH selection schemes. One is *fixed selection scheme*, in which an identical CH will be selected at different times for the queries that have the same query resolution result. The next is *dynamic selection scheme*, in which various CHs will be selected at different times for the queries that have the same query resolution result. Note that for the fixed selection scheme a CH is selected to be the node nearest to the centre location of the query area, whereas for the dynamic selection scheme a rotation operation on the CH is utilised.

Localised data query distribution consists of query unicast and query geocast. The unicast distribution is used to deliver query messages from a sink to the CH. Based on the nodes' location there are many approaches for the sink to calculate a source route to the CH. For example, in the Sink-CHs-Sensors scheme [11], the sink first selects and includes in the source route the sensor that is nearest the CH among one-hop neighbours of the sink. Then the sink calculates the next sensor in the source route, by selecting the sensor nearest to the CH among the neighbours of the previous selected sensors in the source route. This process continues until a source route is found to the CH. As soon as the query message is delivered by the CH in the query area, it is geographically broadcast to all sensors inside the query area. The CH forwards the query message to its one-hop neighbours.

Localised sensing data collection consists of local data delivery, data aggregation, and aggregated data delivery to the sink. On receiving a query message, the corresponding sensors send the sensing data back to the CH in a local region using the reverse path obtained from the query geocast initiated by the CH. The CH collects the sensing data locally before forwarding it to the sink and aggregates the sensing data by placing multiple sensing data into one packet.

In the following an analysis and discussion is provided on the energy bottleneck in data collection of both, the Sink-Sensors-Sink and the Sink-CH-Sensors schemes, by means of the fixed and dynamic selection schemes, suggested above. As it will be seen, under certain conditions, the effective selection of CHs and MHs are comparable over the conventional Sink-Sensors-Sink retrieval model in terms of both balanced and saving energy consumption.

III. Energy Model Analysis

A. Selection of CHs in Data Query

For simplicity we assume the query area is a square consisting of $|S| = s$ nodes. The average energy consumption of a one-hop transmission of a packet is assumed to be E_0. In the conventional model flooding is typically adopted for disseminating a query to sensors in the network. The energy cost of a data query is given by $E_{dq}^c = s * E_0$. In the Sink-CHs-Sensors information retrieval model, the operation consists of unicasting from sink to CH, and geocasting from CH to sensors in the geocast area, which is a combined rectangle area that contains both CH and the query area. Assuming $p \in (0, 1]$ presents the ratio of the number of sensors in the geocast to the total sensors s, then $p * s$ equals the number of sensors in the query geocast area. If S_{hops} denotes the number of hops of transmission in the unicast from the sink to the CH then, the energy cost of Sink-CHs-Sensors query is $E_{dq}^{ch} = p * s * E_0 + S_{hops} * E_0$. As a result, the ratio λ_{dq} of data query cost of the Sink-CHs-Sensors model to the conventional model is given by $\lambda_{dq} = E_{dq}^{ch}/E_{dq}^c = (p * s + S_{hops})/s$, which means that greater energy savings is obtained for smaller values of λ_{dq}. Thus, when $\lambda_{dq} \leq 1$ or equivalently $S_{hops} < s(1 - p)$ a Sink-CHs-Sensors query saves energy compared with the conventional scheme. If a query interest area is defined, both values of p and S_{hops} can be determined by the position of a CH.

B. Selection of CHs in Data Collection

In the Sink-Sensors-Sink conventional scheme of data collection, the energy cost can be approximately given by the sum cost of the replied data unicast from each sensor to the sink. Let $p_1, p_2 \in (0, 1]$ be the ratio of sensors that will reply a query message, and the ratio of aggregated data size to the non-aggregated data size, respectively. Then $p_1 * s$ equals the number of sensors corresponding to the Sink-CHs-Sensors

query and $p_2 * (p_1 * s)$ equals the number of sensors corresponding to the aggregated data size. In addition, let $\bar{L}_0$ and $\bar{L}_{ch}$ be the average route length from each corresponding sensor to the sink and the average route length from each corresponding sensor to the CH, respectively. Then, the energy cost of the conventional data collection scheme, denoted by E^c_{dc}, is given by $E^c_{dc} = p_1 * s * E_0 * \bar{L}_0$, whereas the energy cost of the Sink-CHs-Sensors data collection information retrieval scheme, denoted by E^{ch}_{dc}, is given by $E^{ch}_{dc} = p_1 * s * E_0 * \bar{L}_{ch} + p_2 * (p_1 * s) * S_{hops} * E_0$. Thus, the ratio of data collection cost of Sink-CHs-Sensors to the conventional model will be given by $\lambda_{dc} = E^{ch}_{dc}/E^c_{dc} = (\bar{L}_{ch} + p_2 * S_{hops})/\bar{L}_0$. Obviously, in case $\lambda_{dc} \leq 1$ or equivalently $\bar{L}_0 > \bar{L}_{ch} + p_2 * S_{hops}$, there is energy saving in Sink-CHs-Sensors data collection scheme compared with convention data collection scheme.

IV. Conclusions

In this work we reviewed some interesting clustering based approaches employing data cleansing, data aggregation, and data query in order to extend the lifetime of sensor networks. Further, we have discussed a distributed algorithm for organizing sensors into a hierarchy of clusters with an objective of minimizing the total energy spent in the system to communicate the information gathered by these sensors to the information-processing center (sink). As it was expected we indicated (a rigorous proof remains an open question) that the sensors, which become the CHs in the proposed architecture spend relatively more energy than other sensors because they have to receive information from all the sensors within their cluster, aggregate this information and then communicate to the higher level CHs or the information processing center. However, cluster-based algorithms along with data aggregation and in-network processing can achieve significant energy savings in the sensor networks.

Data aggregation and in-network processing techniques is performed at two levels. We introduced a method to select master/local CHs such that the network lifetime is maximized. Clearly, data aggregation was affected by several factors, such as the placement of aggregation points, the aggregation function, and the density of the sensors in the network. In this framework the determination of an optimal selection of aggregation points, by means of reducing the number of redundant data sent to the end user while preserving data integrity, was crucial and thus very important. In the analysed schemes, sensing data was collected at appropriate CHs, at which data was aggregated and sent to the sink. A query's name was resolved into the IDs and locations of corresponding sensor nodes before being distributed to the network. According to the location of sensor nodes, query distribution and data collection were performed in a corresponding local area. The query message was efficiently unicasted to the CH in a query area, and was then forwarded to a localised area of the network. Sensing data were collected at a CH, at which data were aggregated and sent to the sink. The energy bottleneck analysis and discussion on the criteria of selecting effective CHs show that the discussed schemes were promising.

References

[1] S. Jeffery, G. Alonso, M. Franklin, W. Hong, and J. Widom, "Declarative support for sensor data cleaning," in *Proc. of the 4th IEEE International Conference on Pervasive Computing and Communications (PerCom'2006)*, Piza, Italy, Mar. 2006.

[2] J. Hellerstein, W. Hong, S. Madden, and K. Stanek, "Beyond average: toward sophisticated sensing with queries," in *Proc. of the 2nd International Conference on Information Processing in Sensor Networks (IPSN'2003)*. Berlin, Heidelberg, Germany, EU: Springer-Verlag, 2003.

[3] Y. Zhuang, L. Chen, X. Wang, and J. Lian, "A weighted moving average-based approach for cleaning sensor data," in *Proc. of the 27th International Conference on Distributed Computing Systems (ICDCS'2007)*. Washington, DC, USA: IEEE Computer Society, 2007.

[4] J. Macqueen, "Some methods of classification and analysis of multivariate observations," in *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkley, UC, USA, 1967, p. 281297.

[5] S.Z.Selim and M. Ismail, "k-means-type algorithms: A generalized convergence theorem and characterization of the local optimality," *IEEE Trans. on Pattern Analysis*, vol. 6(1), pp. 81–86, 1984.

[6] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. on Wireless Communications*, vol. 1(4), pp. 660–670, 2002.

[7] K.Padmanabhan and P.Kamalakkannanand, "Energy efficient adaptive protocol for clustered wireless sensor networks," *International Journal of Computer Science Issues (IJCSI)*, vol. 8(5), pp. 296–301, 2011.

[8] O. Younis and S. Fahmy, "Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. on Mobile Computing*, vol. 3(4), pp. 366–379, 2004.

[9] M. Garey and D. Johnson, *Computers and Intractability. A guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[10] R. Haider, M. Javed, and N. Khattak, "Design and implementation of energy aware algorithm using greedy routing for sensor networks," *International Journal of Security and its Applications*, vol. 2(2), pp. 71–86, 2008.

[11] R. Teng, B. Zhang, and Y. Tan, "Design and analysis of intelligent sink for the information retrieval in sensor networks," in *Second International Conference on Sensor Technologies and Applications (SENSORCOMM'08)*, USA, Aug. 2008, pp. 348–353.