IJETCR Open Access

ISSN: 2348 - 2117

International Journal of Engineering Technology and Computer Research (IJETCR)
Available Online at www.ijetcr.org
Volume 3; Issue 2; Page No. 34-40

Data Collection using Artificial Intelligence on Wireless Sensor Networks

Hemkumar D[1], Mohan Gowda V[2], Deepa R[3]

[1]Assistant professor, Gitam University

1986.hem.kumar@gmail.com

[2]Assistant professor, Gitam University

mohangowdav@gmail.com

[3]M.Tech, SJCIT

ABSTRACT

Data collection is a challenging task in wireless sensor networks. In applications of wireless sensor networks, data collection is a wise choice due to the constraints in communication bandwidth and energy budget. This paper focus on efficient data collection with error bounds in wireless sensor networks. The key idea of our data collection approach is to divide a sensor network into clusters, find out local data correlations on each cluster head, and perform global data collection on the sink node based on parameters uploaded by cluster heads. Specifically, we propose a local estimation model to approximate readings of sensor nodes in subsets, and prove rated error-bounds of data collection using local estimation model method. In the process of data collection method, we formulate the problem of selecting the minimum subset of sensor nodes into a minimum dominating set problem which is known to be NP-hard, and greedy heuristic algorithm to find an approximate solution. We use another monitoring algorithm to adjust the composition of node subsets according to changes of sensor readings. Artificial Networks (ANNs) has been used for missing field data recovery. An architecture involve hash technique has been used to remove duplicate data values from a dataset. Finally we demonstrate that data collection remarkably reduces communication cost of data collection with guaranteed error bounds.

Keywords: Data collection, Sensor nodes, artificial neural networks

INTRODUCTION:

A wireless sensor network consists of distributed autonomous sensors to monitor physical or environmental conditions, such as sound, pressure, temperature, etc. and to cooperatively pass their data to a main location through the network. The wireless sensor network is made of "nodes" – from a few to several hundreds or even thousands, where each node is connected to one or sometimes several sensors. Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting.

Recent advances in low-power wireless technologies have enabled wireless sensor networks (WSNs) in a variety of applications, such as coal mine monitoring, scientific observation, and object tracking. Data collection is a challenging task for WSNs, due to the constraints in communication bandwidth and energy budget .On one hand, many applications requires long-term data collection, since the gathered data make sense only if the data collection procedure lasts for months or even years without interruption. On the other hand, sensor nodes are often battery powered and deployed in harsh environments, hence data collection model must be carefully designed to reduce energy consumption on sensor nodes, so as to prolong the network lifetime as much as possible.

In many applications, it is often difficult to continuously collect the complete data from there source-constrained WSNs. From the point of WSNs, large amount of raw data is directly sending to the sink can lead to several problems. First, the data quality may be deteriorated by packet losses due to the limited bandwidth of sensor nodes. Second, intensive data collection incurs excessive communication traffic and potentially results in network congestions. Packet losses caused by such congestions further deteriorate the data quality.

Many of the existing methods are too expensive, i.e., consuming a large amount of computing capacity or

Corresponding author: Hemkumar D

storage capacity .Some methods are too simple to contain enough information. Our approach proves that simplicity and efficiency can be achieved by exploiting implicit sensor node cooperation and elaborately distributing data processing tasks to sensor nodes. The data collection scheme should be self-adaptive to environmental changes. Note that physical environmental changes are usually complex and hard to be modeled comprehensively with a simple estimation model. For long-term data collection, the data collection scheme should be capable of automatically adjusting its parameters according to the environmental changes so as to guarantee its correctness. Data collection method achieves low communication cost by exploiting the fact that physical environments generally exhibit predictable stable state and strong temporal and spatial correlations, which can be used to infer the readings of sensors. Both the scalability and simplicity of data collection are achieved by exploiting implicit cooperation and distributing data processing among sensor nodes. Data collection method can discover local data correlations and suppress the spatial redundancy of sensor data in a distributive fashion. The distributed spatial data correlation discovery and spatial redundancy suppression is achieved by dividing a WSN into several clusters. The sink can estimate the sensor readings according to the model parameters updated by the cluster heads. This distributed data process scheme makes data collection can be easily applied to WSNs with different system scales. As the sensor network scale increases, data collection only needs to increase the number of clusters. Furthermore, by using clustering-based data collection process scheme, sensor data can be processed locally in data collection.

1. Each sensor node is responsible for processing sensor readings generated by it.
2. The spatial redundancy of sensor data is suppressed by cluster heads that are close to the data source.
3. There are no explicitly control data exchange between sensor nodes and their cluster heads.
4. The sensor data process cost is distributed to all sensor nodes and the sensor data process burden of each cluster head can be easily controlled by adjusting the cluster size.

Related Work:

There have been many related works on data collection in WSNs. Directed diffusion is a general data collection mechanism that uses a data-centric approach to choose how to disseminate queries and gather data. Cougar and TinyDB provide query based interfaces to extract data from sensor networks. Those works mainly focus on query-based data gathering, but none of them consider the case of efficient long-term large-scale data collection.

Query-based remote continuously approximate data collection in sensor networks is closely related to the problem we study here. One such approach is approximate caching which gives approximate answers to queries in distributed environments with a fixed error bound. The idea is that the sink uses a constant to reconstruct a piecewise constant approximation of the real sensor readings. No updates are sent until a sensor node notices that its value has diverged by more than a given upper bound from the last reading sent to the sink.

Other query-based approaches extract data from sensor networks by using Gaussian joint distribution to capture the correlations of sensor readings, such as BBQ is the first one using multivariate Gaussian joint distribution to capture the correlations of sensor readings. It samples a small fraction of sensor data from a WSN and utilizes Gaussian joint distribution model to estimate the no sampled sensor readings. Gaussian joint distribution based approaches have several drawbacks that make them unsuitable for long-term large-scale WSNs. First, this kind of models need an expensive long training phase and a complete data set of every sensor node within a sufficiently long period. Gathering complete data set is too energy consuming and even impractical for large-scale WSNs with limited bandwidth. Second, the correctness of this kind of models requires continuous model update which needs periodically gathering the data generated by every sensor node and disseminating the update information to related sensor nodes. Both of the two tasks are costly for energy-constrained WSNs, even when the update frequency is low. Third, it is almost impossible for this kind of models to precisely control the data error. A Gaussian process (GP) is associated with a mean function and a positive-definite kernel function often called the covariance function. An important property of GPs is that the posterior variance of one of its variable depends on the covariance function instead of the actual observed value. Hence, the estimation errors of the no sampled sensor readings are unknown and the estimation quality of the no sampled sensor readings cannot be guaranteed. In comparison, the data processing burdens of Data collection are distributed to each sensor node. The local estimation and the data approximation of data collection are, respectively, settled on each sensor node and each cluster head. This enables sensor data to be processed near or at their sources. The correctness of local estimation is guaranteed by each sensor node locally and the data error bound of Data collection is jointly controlled by the local estimation and the data approximation. No explicit control message exchange is required. And the data error bound of data collection can be flexibly adjusted according to the requirements of applications. Such features make our approach Data collection scalable and efficient for long-term continuous data gathering applications.

Lazaridis and Mehrotra propose to use time-series method to create piecewise linear approximations of signals generated by sensor nodes, and send those approximations to the sink. Their approach gathers a large amount of data and tries to approximate them, rather than exploiting the temporal correlations among sensor readings.

Methodology:

There are mainly three modules, they are:

1. Network deployment

Random network deployment method is used .Because in WSN number of nodes will vary due to their battery backup.

2. Approximate data collection

In this model we are collecting data .this model is sub divided into the local estimation and the data approximation. In the data approximation model we use. Bond Energy Algorithm has been applied for clustering the dataset. Artificial Neural Networks (ANNs) has been used for missing field data recovery. A pass through architecture via hash technique has been used to remove duplicate data values from a dataset.

3. Analysis

In this model information collected during the execution of SAF and ADC algorithms are written to Data Log file and both are compared.

Architecture of Data Collection

The figure 1 shows Architecture of Data Collection components are described below

Local estimation: Sensor node can estimate a newly generated reading through a data model learned from its historic data.

Data model: We use model proposed in An Energy Efficient Querying Framework in Sensor Networks for Detecting Node Similarities. Linear model we can abstract the spatial correlation between different sensors and the error of our spatial correlation model can be easily controlled. This linear data model does not require a large amount of training data or a priori knowledge of the distribution of sensor values.

Parameter Learning: It computes the difference between each reading stored in Q and its estimated trend value.

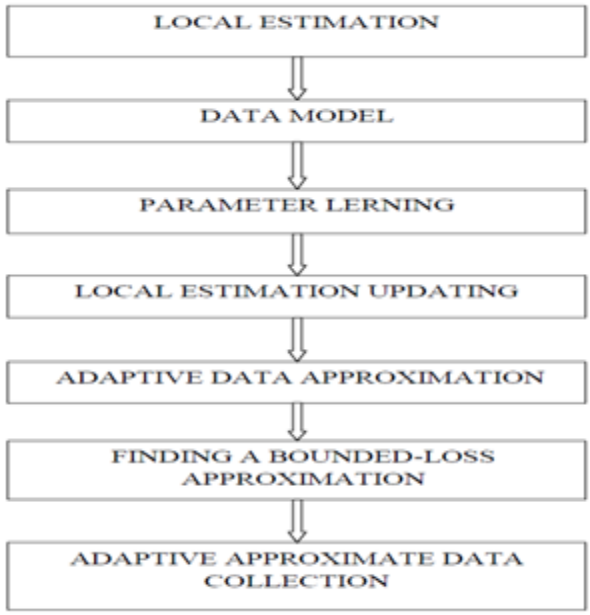


Figure 1: Architecture of Data Collection

Local Estimation Updating: Each sensor node periodically checks the correctness of its local estimation model and updates the parameters of its local estimation as needed as in approximate caching.

Adoptive Data Approximation: Utilizing the data generated by the local estimation of all sensor nodes, the cluster heads cooperate with the sink node to derive a Δ loss approximation.

Finding A Bounded Loss Approximation It makes the data processing close to the data source which can reduce the burden of data transmission**.**

Data Collection: it uses two algorithms

1. The learning algorithm uses the greedy algorithm to find minimum dominating set at Cluster head.
2. The Monitoring algorithm at sink node and cluster head.

Overall solution architecture of the solution is developed which can handle those needs.

General Assumptions and Dependencies

The normal assumption for this research work is the fact that the computing power of clusters has been rapidly growing up towards petascale capability, which requires petascale I/O systems to provide data in a sustained high-throughput manner. Large scale clusters running long running scientific applications use and generate terabytes and in some cases petabytes of data. Satellite imagery, oceanography and a variety of other fields generate petabytes of data, which is must be stored and accessed in a efficient manner.

Development Methods

The development method used in this software design is the modular/functional development method. In this, the system is broken down into different modules, with a certain amount of dependency among them. The input-output data that flows from one module to another will show the dependency. Data flow diagrams have been used in the modular design of the system. Structure charts have been used to show the hierarchy of the function calls in the system.

Data Flow Diagrams

Data-flow models are an intuitive way showing how data is processed by a system. At the analysis level, they should be used to model the way in which data is processed in the existing system. The notation used in these models represents functional processing, data stores and data movements between functions. Dataflow models are used to show how data flows through a sequence of processing steps. The data is transformed at each step before moving on to the next stage. These processing steps or transformation are program functions where dataflow diagrams are used to document a software design. With a dataflow diagram, users are able to visualize how the system will operate, what the system will accomplish and how the system will be implemented. Old system dataflow diagrams can be drawn up and compared with the new systems dataflow diagrams to draw comparisons to implement a more efficient system. Dataflow diagrams can be used to provide the end user with a physical idea of where the data they input, ultimately has an effect upon the structure of the whole system from order to dispatch to restock how any system is developed can be determined through a dataflow diagram. There are several common modeling rules to be followed while creating DFDs are as follows:

- All processes must have at least one data flow in and one data flow out.
- All processes should modify the incoming data, producing new forms of outgoing data.
- Each data store must be involved with at least one data flow.
- Each external entity must be involved with at least one data flow.
- A data flow must be attached to at least one process.

There are three levels of data flow diagrams. These are:

0-Level data flow diagram: It is common practice to draw the 0-level data flow diagram first, which shows the interaction between the system and external agents which act as data sources and data sinks. The figure 2 shows the 0-level data flow diagram, it deploys the network and it collect approximate data in different algorithm and finally it analysis.

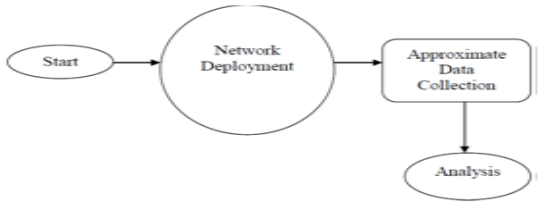


Figure 2: 0th-Level data flow diagram

1st-Level data flow diagram: This context-level Data Flow Diagram is next "exploded", to produce a Level 1 Data Flow Diagram that shows some of the detail of the system being modeled. The below figure 3 shows the 1 level Data Flow Diagram it shows detailed modeling as Approximate Data Collection model is further divided into:

- The Local Estimation
- The Data Approximation

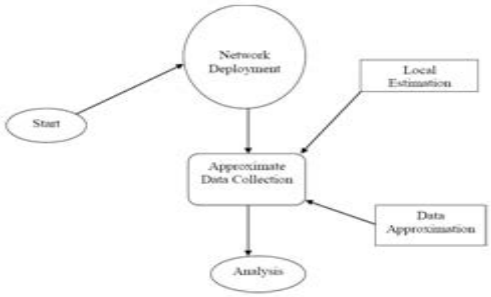


Figure 3: 1st-level data flow diagram

2nd-Level data flow diagram: The Level 2 Data Flow Diagram shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system. The below figure 4 shows the level 2 Data Flow Diagram.

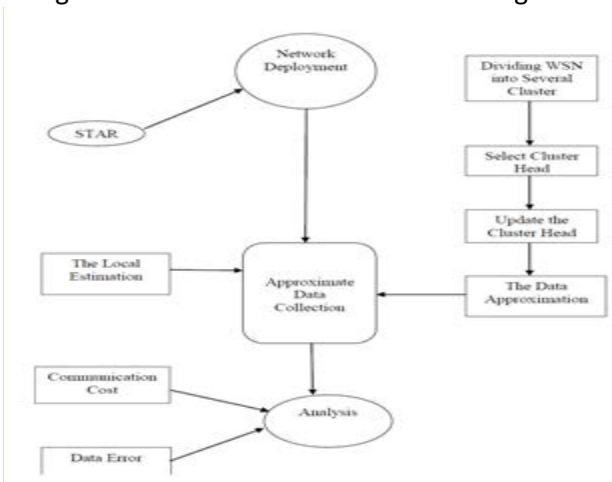


Figure 4: 2-level data flow diagram

Context flow diagram

A Context Diagram in software engineering and systems engineering are diagrams that represent all external entities that may interact with a system. This diagram is the highest level view of a system, similar to Block Diagram, showing a, possibly software based, system as a whole and its inputs and outputs from/to external factors. System Context Diagram is diagrams used in systems design to represent all external entities that may interact with a system.

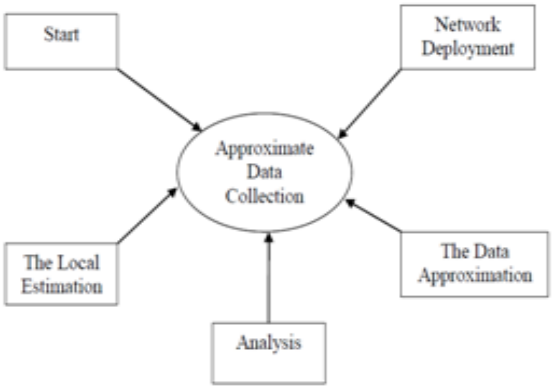


Figure 5: Context flow diagram

This diagram pictures the system at the center, with no details of its interior structure, surrounding by all its interacting systems, environment and activities. The objective of a system context diagram is to focus attention on external factors and events that should be considered in developing a complete set of system requirements and constrains. System context diagram are related to Data Flow Diagram, and show the interactions between a system and other actors with which the system is designed to face. System context diagrams can be helpful in understanding the context in which the system will be part of software engineering. Context diagrams are used early in a project to get agreement on the scope under investigation. Context diagrams are typically included in a requirements document. The diagram of the overall project work is displayed above in Figure 5. In figure 5, A WSN is created by drawing nodes in network. Modules are created to support creation of nodes, approximate data collection, local estimation and analysis.

Results:

Our work detects data similarities among the sensor nodes by comparing their local estimation models rather than their original data. The simulation results show that our approach can greatly reduce the amount of messages in wireless communications by as much as 21 percent compared with existing works. The below figure 8 shows that snapshot for data collection in wireless sensor network, it includes network configuration, routing information, algorithm selection, simulation control and status of simulation. And each window used many attributes to specify the values. of configuration. i.e., for network configuration window includes network size, sensor radius, sensor period, sensor cost, data transaction period, data transaction cost, receive cost and transmission radius. If we insert the entire attribute values in all windows and then choose existing algorithm (Similarity-Based Adaptable Framework), network deployed for existing system as showed below figure 6.

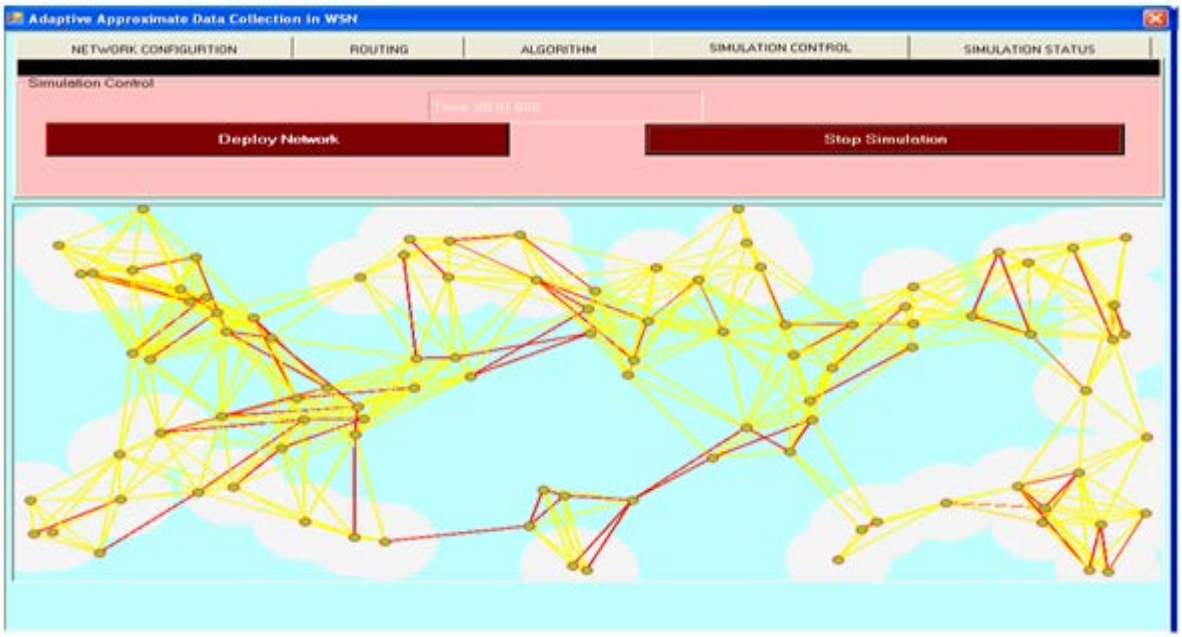
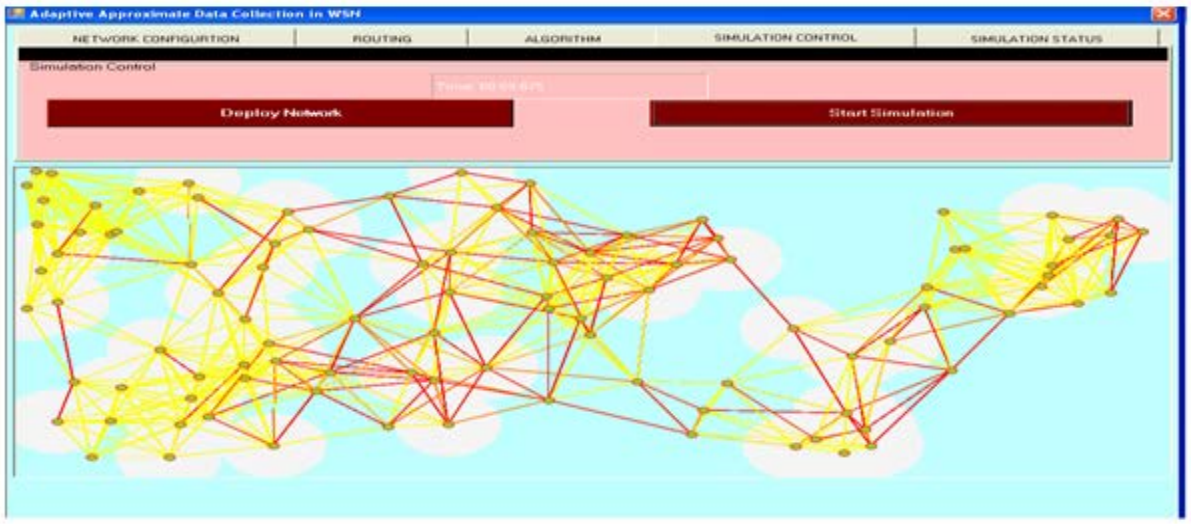


Figure 6: snapshot for data collection in wireless sensor network

On other hand, we choose proposed system, the network deployed for proposed algorithm showed below figure.



The table showed below is Compression of Data Error and Communication Cost between the existing system and proposed system

Name	NW Size	Power Remaining(mJ)	Est Error Bound	Cost(mJ)	Data Aggregrated(Packets)
ES	100	91900	0.450490196	204	90458
ES	100	81700	0.216710875	377	72469
ES	100	69810	0.124217082	562	45691
ES	100	58220	0.078675676	740	10297
PS	100	95390	0.836754386	114	94754
PS	100	87040	0.344031621	253	83071
PS	100	78320	0.223771429	350	67038
PS	100	69200	0.153777778	450	46949

The following figure 7 shows that graph of power remaining between the existing system and proposed system.

The following figure 7 shows that graph of power remaining between the existing system and proposed system.
The below figure 8 shows that graph of cost between the existing system and proposed system
The simulation results show that our approach can greatly reduce the amount of messages in wireless communications by as much as 21 percent compared with existing works

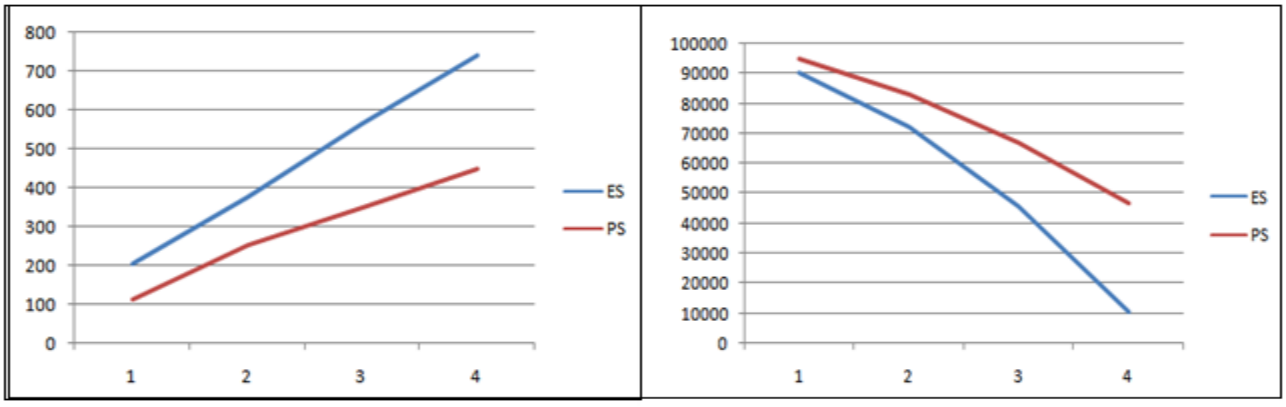


Figure 7: power remaining between existing system and proposed system. Figure 8: Cost between existing system and proposed system.

CONCLUSION:

The propose a novel data collection strategy in WSNs. Data collection can approximate all readings of a sensor network by exploiting the fact that physical environments frequently exhibit predictable weak stable state and strong temporal and spatial correlations between
sensor readings. Our work detects data similarities among the sensor nodes by comparing their local estimation models rather than their original data. The simulation results show that our approach can greatly reduce the amount of messages in wireless communications by as much as 21 percent compared with existing works.

REFERENCES:

1. G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A Macroscope in the Red Woods," Proc. (SenSys '05), 2005.
2. L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, and G. Dai, "Canopy Closure Estimates with GreenOrbs: Sustainable Sensing in the Forest," Proc. Seventh ACM Conf. (SenSys '09), 2009.
3. D. Chu, A. Deshpande, J.M. Hellerstein, and W. Hong, "Approximate Data Collection in Sensor Networks Using Probabilistic Models," (ICDE '06), 2006.
4. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-Driven Data Acquisition in Sensor Networks," Proc. 13th Int'l Conf. Very Large Data Bases (VLDB '04), 2004.
5. Wan, S.B. Eisenman, and A.T. Campbell, "Coda: Congestion Detection and Avoidance in Sensor Networks," Proc. First Int'l Conf. Embedded Networked Sensor Systems (SenSys '03), 2003.
6. Guestrin, P. Bodi, R. Thibau, M. Paski, and S. Madde, "Distributed Regression: An Efficient Frame Work for Modeling Sensor Network Data," Proc. Third Int'l Symp. Information Processing in Sensor Network (IPSN), 2004.
7. Adaptive Approximate Data Collection for Wireless Sensor NetworksChao Wang, Huadong Ma, Member, IEEE, Yuan He, Member, IEEE, and Shuguagn Xiong