A Dynamic Clustering and Scheduling Approach to Energy Saving in Data Collection from Wireless Sensor Networks

Chong Liu
Computer Science Dept.
University of Victoria
BC, Canada V8W 3P6
chongliu@cs.uvic.ca

Kui Wu
Computer Science Dept.
University of Victoria
BC, Canada V8W 3P6
wkui@cs.uvic.ca

Jian Pei
School of Computing Science
Simon Fraser University
BC, Canada V5A 1S6
jpei@cs.sfu.ca

Abstract— **Energy consumption is one of the major constraints in wireless sensor networks. A highly feasible strategy is to aggressively reduce the spatial sampling rate of sensors (i.e., the density of the measure points in a field). By properly scheduling, we want to retain the high quality of data collection. In this paper, we propose a novel *dynamic clustering and scheduling approach*. Orthogonal to most existing methods which mainly utilize the overlaps of sensing ranges of sensors to reduce the spatial sampling rate, our method is based on a careful analysis of the surveillance data reported by the sensors. We dynamically partition the sensors into groups so that the sensors in the same group have similar surveillance time series. They can share the workload of data collection in the future since their future readings may likely be similar. A generic framework is developed to address several important technical challenges, including how to partition the sensors into groups, how to dynamically maintain the groups, and how to schedule sampling for the sensors in a group. We conduct an extensive empirical study to test our method using both a real test bed system and a large-scale synthetic dataset.**

I. Introduction

A wireless sensor network may consist of a large number of sensor nodes, and each node is equipped with sensors, microprocessors, memory, wireless transceiver, and battery. Once deployed, the sensor nodes form a network through short-range wireless communication. They collect environmental surveillance data and send them back to the data processing center, which is also called the *sink node*.

In many applications, wireless sensor networks are used to monitor some measures of interest, such as temperature, light intensity, air pressure, etc. In order to obtain accurate surveillance, *spatially frequent sampling* is required to capture the variation of a monitored measure. That is, the density of the measure points in a field should be sufficiently high so that the spatial distribution of the monitored measure can be restored with high quality.

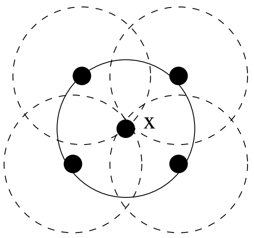

Fig. 1. Redundant Sensors Based on Sensing Ranges.

However, the energy consumption is one of the major constraints in wireless sensor networks. Typically, the capacity of the battery in a sensor node is limited. Moreover, due to the sheer number of sensor nodes and the potentially hostile environment, it is usually very hard, if not impossible at all, to recharge the battery after the deployment of a sensor node. How to extend the lifetime of a sensor network under the stringent energy constraint of each individual sensor node is an important and challenging problem. Our goal is to maintain the satisfactory spatial sampling rate for data collection and extend the lifetime of the sensor nodes as long as possible.

A critical observation is that all sensor nodes reporting data may generate a lot of redundant data. For example, in Figure 1, if the sensing range of a sensor node x is fully covered by the sensing ranges of some of its neighboring sensors, then sensor x can be regarded redundant. Stimulated by this observation, most existing methods [8], [18], [19], [20] adopt the *coverage-based scheduling methods*. That is, they check the redundancy based on the sensing coverage of sensor nodes. In this case, the pairwise geographic distance between sensor nodes is the only factor considered in sensor scheduling, and the output of the scheduling algorithm is a scheduling scheme that samples the field uniformly in space.

The coverage-based scheduling methods may not work well for heterogeneous environment where the monitored measure changes quickly in some sub-regions but slowly in others. It is also possible that the readings of two geographically proximate sensor nodes are dramatically different due to a boundary that separates the two sensor nodes into two sub-regions with distinguished features. For example, if the temperature is monitored, two proximate sensor nodes can report very different values if one is directly under sunshine and the other one is in the shadow. Furthermore, it is often very difficult to accurately estimate the sensing range of a sensor node, since it is highly related to the local environment in which the sensor node is deployed. As a result, the scheduling purely based on the sensing coverage may not be efficient or effective.

The major inherent problem of the coverage-based scheduling methods is that *they only rely on the static structure of the sensor networks and are not aware of the data reported by the sensor nodes.* Intuitively, the correlation between the data reported by the sensor nodes may help to reduce the spatial sampling rate of the sensor nodes substantially.

Example 1—Intuition: The readings reported by a sensor node over time form a time series. Suppose the time series of sensor nodes x, y, and z are very similar in the past. Thus, we may conjecture that the readings of x, y, and z would also likely be similar in the future. Thus, instead of scheduling the three sensor nodes reporting data simultaneously, we can let two out of the three sensor nodes report at a time and all the three take turn to report. Such a schedule has the following two advantages.

- *Energy saving.* Each sensor node saves 33.3% energy on reporting data.
- *Quality guarantee.* When the three sensors are still correlated, we can obtain the data with high quality and also save energy. On the other hand, even if the readings of one sensor node is not similar to the other two, we still can detect the divergency with a minor delay. Then, an updated schedule can be made based on the change. ■

Motivated by the intuition in Example 1, in this paper, we develop a dynamic clustering and scheduling approach to solve the typical data collection problem in wireless sensor networks: how the sink node can collect data from highly-redundant geographically-distributed sensor nodes with high observation fidelity and low energy consumption. Our solution is to dynamically group sensor nodes into a set of disjoint clusters such that the sensor nodes within a single cluster have current strong spatial correlation, hence great similarity in observations. Therefore, all the sensor nodes in a cluster can be treated equally, and at any time instance, only *one* sensor node is needed to be active, serving as the representative for the whole cluster. All the rest of sensor nodes can go to sleep without much degradation of observation fidelity, since at the presence of the representative node, other nodes will not bring much gain. To balance the workload, the sensor nodes within a cluster can be scheduled to work in a round-robin way.

The clustering operation is based on the dissimilarity measure of time series consisting of historical observations from individual sensor nodes. The degree of spatial correlation can be evaluated by the dissimilarity measure. For two locations with high spatial correlation, their corresponding time series are usually associated with a low dissimilarity measure. Therefore, in a very smooth sub-region, the observed measure has only small changes within the sub-region, that is, the difference between observations at any two locations within the sub-region may be quite small. Hence, the working sensor nodes within this sub-region could be sparse without losing observation fidelity. In contrast, in a not so smooth sub-region, the working sensor nodes should be dense. By setting an appropriate dissimilarity measure threshold value to distinguish similar nodes from dissimilar nodes, the spatial sampling rate will match the spatial variation of the observed physical phenomenon. A smaller threshold value increases the spatial sampling rate and observation fidelity, but it requires more energy consumption. In this sense, the dissimilarity measure threshold value constitutes a degree of freedom that could be tuned to balance the trade-off between observation fidelity and energy consumption.

Another advantage of our method is that, *the difficulty of estimating a sensor's sensing range is avoided* because the dissimilarity measure relies solely on the data reported.

While some research exploits the spatial correlation among sampling data, the correlation is used in the context of in-network aggregation [15], where the redundant data are suppressed in some aggregation nodes before they are transmitted to the sink node. This paper utilizes the spatial correlation in the context of sensor scheduling. With our method, data redundancy among sensor nodes can be determined in a run-time system to avoid the generation and transmission of redundant data. So we can expect a much higher energy saving than in-network data aggregation. Note that in addition to exploiting spatial correlation, exploiting temporal correlation among sensor nodes' observation can further reduce the energy consumption. Nevertheless, due to the limit of space, we only consider spatial correlation in this paper.

The rest of the paper is organized as follows. In Section II, we propose an Energy Efficient Data Collection (EEDC) framework. We introduce the key modules of EEDC, namely sensor node clustering and sensor node scheduling, in Sections III and IV respectively. EEDC is then evaluated with a real test bed using MICA2 sensors [5] and also with a large-scale synthetic dataset in Section V. We review related work in Section VI and finally conclude the paper in Section VII.

II. The Energy Efficient Data Collection (EEDC) Framework

Compared to sensors nodes, the sink node usually has much larger memory and much powerful computing capability. Also, the energy supply is generally not a big problem for the sink node. Such an asymmetry between the sink node and the sensor nodes determines that a good design for data collection should not put heavy burdens on sensor nodes. Instead, the heavy duties should be assigned to the powerful sink node. Our Energy Efficient Data Collection (EEDC) framework follows this design principle and is shown in Figure 2. As we can see, the functionalities in sensor nodes are much simpler than those in the sink node. In a sensor node, the scheduler module simply extracts the working schedules received from the sink node and makes the sensor node work/sleep according to the schedule. In contrast, the sink node takes most workloads, including four main functional modules as shown in Figure 2.

The framework has the following major modules.

1) *The data storage module.* It stores all sampling data received from the sensor nodes. This module records a time series for each sensor, which is fed into the dissimilarity measure module as input data.
2) *The dissimilarity measure module.* It calculates the pairwise dissimilarity measure of time series. Dissimilarity measure is application specific, and it is impossible to use a common dissimilarity measure to accommodate all application scenarios. As such, this model has an input parameter from the user, specifying the criterion for dissimilarity measure for a specific application scenario. We will introduce the details of dissimilarity measure used in our experimental study in Section V-B.
3) *The clustering module.* Given the dissimilarity computed by the dissimilarity measure module and a maximal dissimilarity threshold value max_dst, this module divides the sensor nodes into clusters, such that the dissimilarity measure of any two sensor nodes within a cluster is less than max_dst. The details of the clustering algorithm will be discussed in Section III.
4) *The sensor node working schedule generator.* It generates a working schedule for each sensor node based on the clusters obtained from the clustering module. The details of sensor scheduling will be discussed in Section IV.

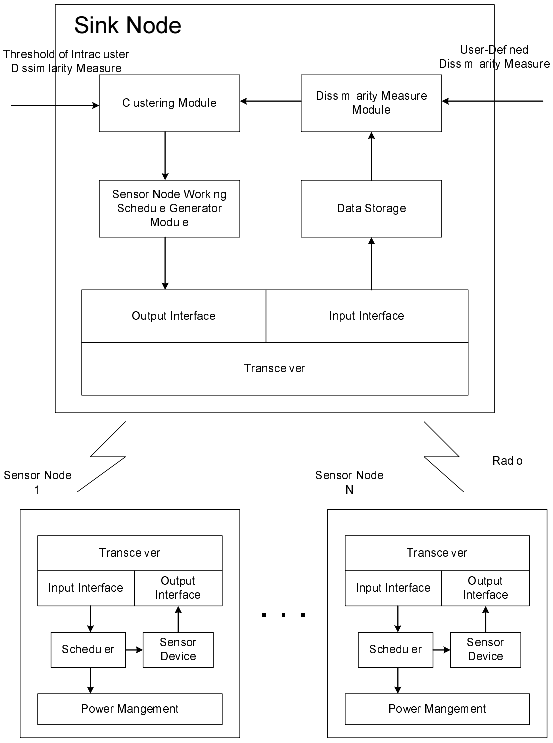

Fig. 2. Energy Efficient Data Collection (EEDC) Framework

With the EEDC framework, the data collection procedure in a sensor network could be divided into the following three phases:

1) *Data accumulation.* In this phase, each sensor node keeps sampling and transmitting samples to the sink node. The sink node records the time-ordered sampling data and maintains a time series for each sensor node. After collecting enough data, the sink node calculates the dissimilarity measure between any two time series. It terminates this phase whenever the dissimilarity measure among the collected time series remains roughly stable.
2) *Clustering.* In this phase, the sink node uses a clustering algorithm to separate sensor nodes according to the dissimilarity measure calculated in the first phase. The output of the clustering algorithm is a set of clusters, and inside each cluster the dissimilarity measure between two arbitrary sensor nodes is smaller than a given threshold value. Consequently, the whole field is divided into pieces of smooth sub-

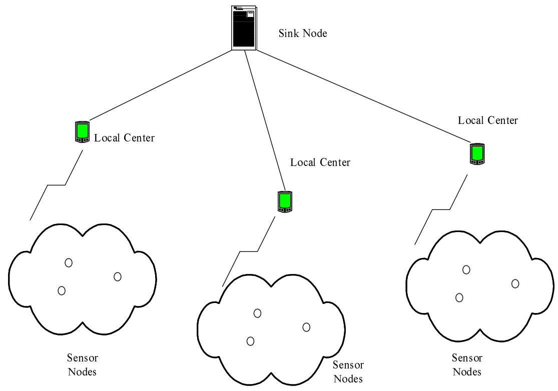

Fig. 3. Energy Efficient Data Collection (EEDC) Framework in a Hierarchical Architecture

regions, each of which is covered by a corresponding cluster of sensor nodes. The observation at any point in this sub-region can be approximated by the observation of any sensor node within the cluster covering this sub-region.

3) *Saving and dynamic clustering*. In this phase, the sink node sends out the decision of clusters to all sensor nodes and requires the sensor nodes within the same cluster to work in turn to save energy. In the mean time, the sink node monitors large variations within a cluster and dynamically adjusts the cluster.

Note that after clustering, the environment may change and thus the previous clusters may not be valid anymore. It is desirable to adaptively change the clusters according to the changes of the monitored measure. The details of re-clustering will be discussed in Section IV when we introduce the scheduling algorithm.

As can be seen, the sink node may become a bottleneck for large-scale networks. When a network becomes very large, the EEDC framework can be easily extended to a hierarchical architecture as shown in Figure 3. A commonly used strategy is to divide the network into several sub-networks, with each depending on a local center for local data collection and long-distance radio transmission to the sink node. In this case, a local center, with the above four main components installed, takes the full responsibility for collecting data, calculating pairwise dissimilarity measure, executing the clustering algorithm, and generating working schedules, within the local region. The sink node only needs to collect information directly from those local centers.

Note that in this paper, we assume the single-hop network architecture, i.e., all the sensor nodes are within single-hop radio transmission to the sink node, or to a local center. With this setting, we do not need to consider network partitioning due to sensor scheduling.

Compared to other existing sensor scheduling schemes that rely on sensing coverage and the static geographic distribution of sensors [8], [18], [19], [20], EEDC distinguishes itself from existing schemes in that it is data-aware and makes scheduling decisions according to the spatial distribution of the monitored phenomenon.

III. Clustering Sensor Nodes

Given the pairwise dissimilarity values, we need a clustering algorithm to partition the sensor nodes into exclusive groups (called *clusters*) such that within each cluster, the pairwise dissimilarity measure of the sensor nodes is below a given intra-cluster dissimilarity threshold max_dst. All sensor nodes in the same cluster are correlated. In each cluster, only one sensor node needs to work at any time instant. Because of this reason, it is desirable to minimize the number of clusters to maximize the energy saving.

Interestingly, the above problem could be modeled as a clique-covering problem. We construct a graph G such that each sensor node is a vertex in the graph. An edge (u, v) is drawn if the dissimilarity measure between vertex u and vertex v is less than or equal to the given intra-cluster dissimilarity measure threshold max_dst. Clearly, a cluster is a clique in the graph. Then, the clustering problem is to use the minimum number of cliques to cover all vertices in the graph.

The clique-covering problem is proven to be NP-complete and even does not allow constant approximation [13], [21]. Hence we propose a greedy algorithm as described in Figure 4 to obtain a rough approximation. The basic idea of the algorithm is to heuristically find cliques that cover more vertices that have not been clustered. Heuristically, the vertices with larger degrees may have a better chance of appearing in larger cliques. Thus, the search starts from the vertex with the largest degree, until all vertices are covered. The output of this algorithm is a set of cliques that covers all vertices.

IV. Scheduling Sensor Nodes

A. Round-Robin Scheduling Based on Clusters

Since all the sensor nodes in the same cluster are considered correlated, it is desirable to schedule them to work alternatively to save energy. For a cluster with k sensor nodes, we can divide a time period T into k time slots, and require only 1 sensor node to work for each time slot. Other sensor nodes in the same cluster can fall asleep. If the sensor nodes work with a round robin scheduling, each

```
Input: a graph G;
Output: a set of cliques covering the graph G;
Algorithm Description:
  Label all vertices in the graph G as uncovered;
  while (there are vertices uncovered in the graph G){
      Pick up the vertex v with the highest node degree among the uncovered vertices;
      Pick up all the vertices adjacent to v and put them into a list of S;
      Construct a graph $G_{tmp}$, consisting of only the vertices in S;
      Calculate the node degree of each vertices in $G_{tmp}$;
      Sort the vertices in S according to the decreasing order of node degree in $G_{tmp}$;
      (To break a tie, the vertex with lower degree in the original graph G precedes);
      Construct a clique C containing only v;
      while (there are vertices available in S){
            Pick up next vertex s from S;
            If s is adjacent to all vertices in C thus far, put s into the clique C;
      }
      Output  clique C;
      Remove all vertices covered by C from the graph G;
  }
```

Fig. 4. The Greedy Algorithm

sensor node needs to work only for a duration of $\frac{T}{k}$ in every time period of T. Note that if we assume that each time slot has a fixed length, different clusters may have different T values and the duration of T in a cluster is proportional to the size of the cluster.

To generate a working schedule for a cluster with k sensor nodes, the sink node randomly selects a working order of the k sensor nodes and then sends the working schedule to the sensor nodes. An example of working schedule is shown in Figure 5.

To make the scheduling algorithm work correctly, sensor nodes should be time synchronized. Time synchronization is one of the fundamental requirements in sensor networks, because large time asynchrony will generate sampling data with incorrect timestamps and any surveillance based on such incorrect information is likely unreliable. To avoid potential monitoring gap due to small time asynchrony, we can simply require that each sensor works a little bit longer at the end of its working shift so that the consecutive sensor in duty has a minor monitoring overlap.

B. Dynamic Adjustment

The environment being monitored by the sensor network might change, and thus the previous clusters might not be valid any more. A good scheduling algorithm should accommodate such changes and dynamically adjust the clusters. This requires the sink node to detect whether the sensor nodes in the current clusters are still correlated, and adjust the clusters according to the changes as quickly as possible. To achieve this, the sink node can utilize the fact that the dissimilarity measures from some sensor nodes in the same cluster should be larger than the given intra-cluster dissimilarity measure threshold if the current cluster does not match the features of the monitored phenomenon.

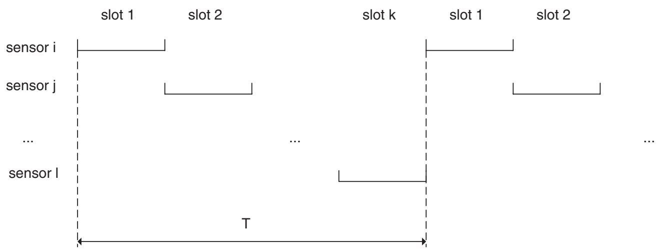

Fig. 5. Round-Robin Scheduling Scheme

To get a quick detection of changes in the spatial correlation, we can extend the working time of each sensor node by Δ_t at the end of its working shift for the purpose of clustering validation. As shown in Figure 6, during the time period of Δ_t, the sink node should receive sampling data from two sensor nodes assigned to two consecutive time slots. By calculating the dissimilarity measure of the two sensors in the time period of Δ_t, the sink node can detect the large variation within the cluster. *The selection of the value of Δ_t is empirical and application specific.*

Remark: If a cluster must be split into two clusters due to the changes of the monitored environment, the changes must be detected by the sink node within the time period of T, even if the sink node only compares sampling data

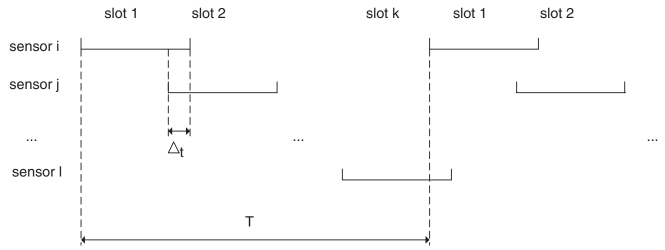

Fig. 6. The Extension of Round-Robin Scheduling Scheme

from two sensor nodes in consecutive time slots.

To prove the above claim, we assume that in real applications, the precondition of splitting an existing cluster is that the changes of the monitored phenomena should form a boundary cutting the current region covered by the cluster. As shown in Figure 7, assume that initially 10 sensors in the rectangular area have similar observations and thus belong to the same cluster. Due to the phenomenon changes, a boundary is formed to cut the rectangular area into two sub-areas, A and B. If all the 10 sensors fall within either A or B, the phenomenon changes cannot be detected and thus there is no need to split the current cluster. If there is at least one sensor in each sub-area, then no matter what the current working order of the 10 sensors is, at least two sensors belonging to different sub-areas must be assigned into two consecutive time slots. Their large dissimilarity will be detected within one round of scheduling time, T. ■

Once the sink node detects that a cluster includes sensor nodes with dissimilarity measure larger than the given intra-cluster threshold value, it asks all sensor nodes in the corresponding cluster to work simultaneously. Then the clustering algorithm will be executed to re-group these sensor nodes to several clusters as a response to local phenomenon changes. It is obvious that the number of clusters will keep increasing, since there is only splitting operation in the above adjustment. In the worst case, most sensors in the network will be woken up to work simultaneously. To avoid this situation, the sink node can re-cluster the whole network when the current number of clusters becomes significantly larger than the number of clusters at the previous network-wide clustering.

When the spatial correlation within a sub-region remains stable, the frequency of dynamic adjustment of clusters should be very low. We stress that *spatial correlation is quite stable for some applications even if the monitored phenomenon changes dramatically*. This observation is demonstrated true in our later experimental study.

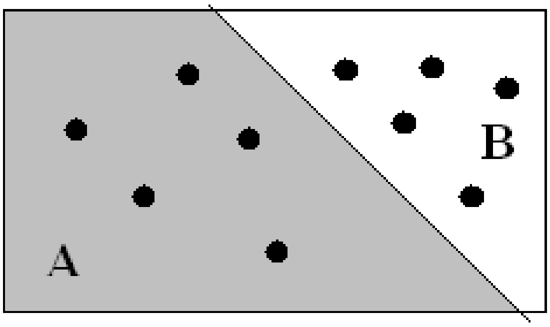

Fig. 7. A Scenario in Which a Cluster Should Be Split

C. Further Discussion

In addition to the above round-robin scheduling method, which maintains a single active sensor node within each cluster, we are currently investigating different scheduling schemes to maintain multiple active sensor nodes per cluster. Due to the limit of space and for the simplicity of this paper, we do not include different variations of the round-robin scheduling method.

Maintaining multiple active sensor nodes per cluster has two obvious advantages. First, multiple active sensor nodes can improve data reliability. In the case of one active sensor node per cluster, the sink node has no way to restore the readings of the sensor nodes in a cluster if a packet from the only active sensor node in the cluster is lost. If multiple active sensor nodes per cluster are used, the data can be restored as long as at least one packet out of the multiple active sensor nodes reaches the sink node. Second, multiple active sensor nodes can help to shorten the delay of cluster split detection and make the system quickly respond to the spatial correlation changes among sensor nodes within a single cluster.

Nevertheless, it is clear that the above benefits are obtained with extra energy consumption. We are again met with the trade-off between observation fidelity and energy consumption.

V. Performance Evaluation

A. Experiment Setup

We experimentally tested the EEDC framework based on MICA2 sensor nodes [5]. As illustrated in Figure 8, we deployed 18 MICA2 sensor nodes in a 3×6 grid layout on a big table to sample the light intensity. A desk lamp with a dimmer was the only light source in the room. We used several boxes with different sizes of holes on the top to divide the area into sub-regions with different light intensity. The deployment of sensor nodes and the boxes are illustrated in Figure 9. The monitored phenomenon was generated by varying the light intensity of the lamp. We

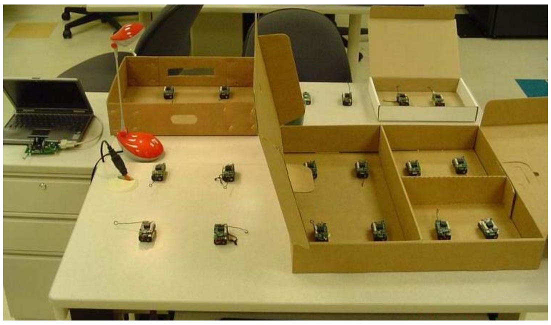

Fig. 8. The Test Bed

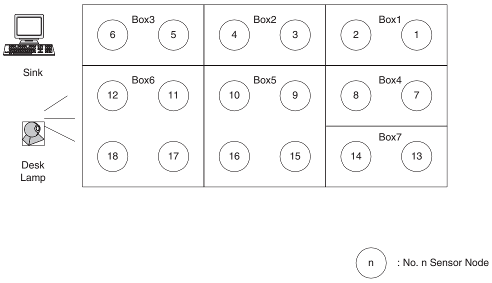

Fig. 9. The Sensor Nodes and The Boxes

implemented the EEDC framework on the sink node and the MICA2 sensor nodes to collect the light intensity data.

The purpose of this experiment is twofold. First, we need to verify the correctness of the proposed clustering algorithm. Second, since the energy saving should not be achieved at the cost of observation fidelity, we need to verify that the information loss rate with EEDC is acceptable.

Although this experiment is not a real application, we remark that the experimental design may be representative for some real applications with sensor networks, for instance, the monitoring system for storage rooms in a grocery warehouse.

B. Calculating Dissimilarity

As described above, the time-ordered data sequence at each sensor node forms a time series, and the clustering algorithm is based on the dissimilarity between the time series of the sensor nodes.

Nevertheless, dissimilarity measure in practice is highly application specific. For example, Figure 10 illustrates two time series of light intensity collected at two separate locations in our experimental test bed with a rate of 2 samples per second. If the magnitude is the main concern, the two time series are not similar at all. On the other hand, if the trends and the patterns are the main concern, they are quite similar.

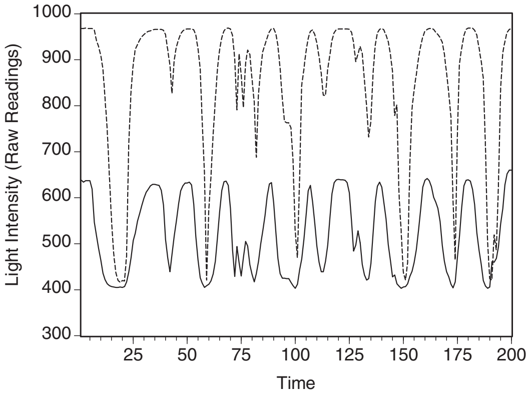

Fig. 10. An Example of Dissimilarity Measure

We define the following two metrics to describe the average difference of two time series in magnitude and in trend respectively.

Definition 1: magnitude m-similar. Two time series $X\{x_1, x_2, ..., x_n\}$ and $Y\{y_1, y_2, ..., y_n\}$ are magnitude m-similar if

$$\frac{\sum_{i=1}^{n} |x_i - y_i|}{n} \leq m.$$

Definition 2: trend t-similar. Two time series $X\{x_1, x_2, ..., x_n\}$ and $Y\{y_1, y_2, ..., y_n\}$ are trend t-similar if

$$\frac{n_1}{n} \geq t,$$

where n_1 is the total number of pairs (x_i,y_i) in the time series that satisfy $\nabla x_i \times \nabla y_i \geq 0, \nabla x_i = x_i - x_{i-1}, \nabla y_i = y_i - y_{i-1}, i > 1$.

In our experiment, we assume that users are mainly interested in the general trend as well as the magnitude of time series. Moreover, if two sensor nodes are spatially far away, even if their readings in the past are similar, the conjecture that their readings will correlate in the future is likely unreliable. Thus, we constrain that the geographic distance between any two sensor nodes that are considered similar must be at most $gmax\_dist$, where $gmax\_dist$ is a given maximal distance threshold. This requirement is also to facilitate the transmission of scheduling decisions from the sink node.

In general, we want the dissimilarity measure to have a straightforward and practical meaning. In this specific experiment, we separate two time series into different groups if *any* of the following constraints is violated:

1) They have a small difference in magnitude on average;
2) They have the same trends in most of time;
3) They are geographically close.

Assume that $gdst(S_x, S_y)$ denotes the geographical distance between sensor node S_x and sensor node S_y, and assume that $gmax\_dst$ is the user-defined threshold value for geographical distance. In our experiment, we put two time series X and Y into different groups if (a) they are not magnitude m-similar, or (b) they are not trend t-similar, or (c) $gdst(S_x, S_y)$ is greater than $gmax\_dst$.

In the following experiment, we calculated the dissimilarity measure within a modest timeframe of 5 minutes. This value was selected depending on the fact that the dissimilarity measure in our experiment remained roughly stable within the time period of 5 minutes and using a relatively larger value did not exhibit big differences.

C. Experimental Results

1) The Correctness of Clustering with EEDC: Although the clique-covering problem for a general graph is NP-complete, it is easy to know the optimal clique covers in our experiment since the knowledge of which sensor node belongs to which sub-region is known as a priori. We use the above criteria to check the dissimilarity and set $m = 30, t = 95\%, gmax\_dst = 3$ feet. By calculating the pairwise dissimilarity measure, we get a graph as shown in Figure 11, where a link between two nodes indicates that they are similar according to the above criteria. From this figure we can see that all cliques consist of the sensor nodes in the same box, which validates the effectiveness of the dissimilarity measure. The output of the clustering algorithm is illustrated in Figure 12, which is apparently the optimal solution for this specific simple graph.

We want to stress again that *spatial correlation is quite stable for some applications even if the monitored phenomenon changes dramatically*. In our experiment, we changed the light strength very quickly by tuning the dimmer of the desk lamp. The sampling data from the sensors within the same box remained similar no matter how fast we changed the light.

2) The Observation Fidelity with EEDC: In order to evaluate the observation fidelity with EEDC, we used the joint entropy of multiple sources [4] to measure the total information obtained. For comparison purpose, we required that all sensors work simultaneously to obtain the best case of lossless information collection. We also applied our clustering and scheduling algorithms on the same dataset to get the information collected with EEDC.

In this case study, we investigated the light intensity data collected in our MICA2 testbed with 18 MICA2 sensor nodes deployed in the way illustrated in Figure 9. The sampling rate was 2 samples per second, and the data collection time was 10 minutes. The onboard ADC translates a light intensity raw reading into a discrete integer value between 0 and 1024.

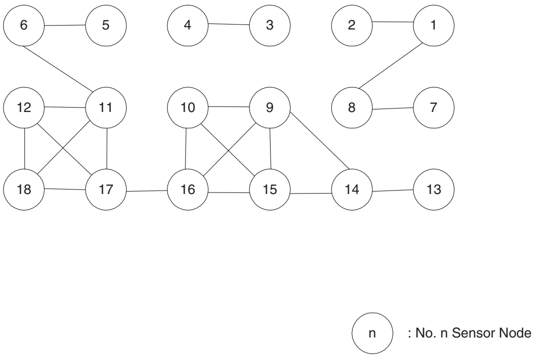

Fig. 11. The Generated Graph

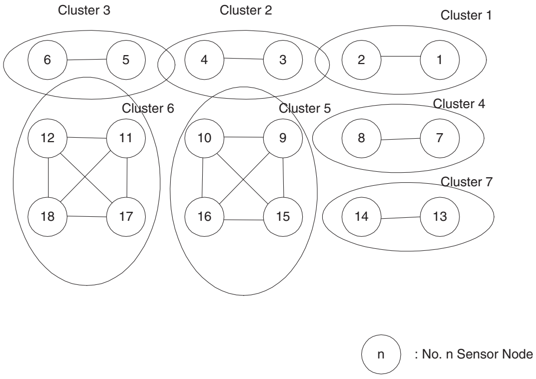

Fig. 12. The Clustering Result with the Real Dataset

We treated the light intensity data from the i-th sensor node as an independent discrete random variable S_i. The joint entropy of the total 18 sensor nodes' observations $H(S_1, S_2, ..., S_{18})$, i.e., the maximum possible total information, can be calculated as,

$$H(S_1, S_2, ..., S_{18}) =$$

$$-\sum_{S_1, S_2, ..., S_{18}} p(S_1, S_2, ..., S_{18}) \times \ln p(S_1, S_2, ..., S_{18})$$

However, calculating $H(S_1, S_2, ..., S_{18})$ in an 18-dimensional space is quite computationally expensive. Instead, we calculated its upper bound. If $H(C_i)$ denotes the joint entropy of all sensor nodes belonging to cluster C_i, then

$$H(C_i) = H(S_a, S_b, ..., S_z) | S_a, S_b, ..., S_z \in C_i$$

In our real dataset collected in 10 minutes with a rate of 2 samples per second, we had

$$H(S_1, S_2, ..., S_{18}) \leq \sum_{i=1}^{7} H(C_i) = 39.7nat.$$

Let X_i denote the current working node of cluster C_i with EEDC. Since only one sensor node in a cluster is required to work in any given time, the expectation of the joint entropy of working nodes, $\overline{H}(X_1, X_2, ..., X_7)$, is the information gathered with EEDC on average. Tested with the real dataset, it can be lower bounded by

$$\overline{H}(X_1, X_2, ..., X_7) \geq \sum_{i=1}^{7} \overline{H}(X_i) - \sum_{i=1,j=1,i\neq j}^{i=7,j=7} \overline{I}(X_i, X_j)$$
$$= 36.5nat,$$

where $\overline{I}(X_i, X_j)$ is the expectation of mutual information of X_i and X_j, and $\overline{H}(X_i)$ is the expectation of the entropy of X_i.

As the result, the actual information loss rate with EEDC in our case study, r_{EEDC}, can be calculated as

$$r_{EEDC} = 1 - \frac{\overline{H}(X_1, X_2, ..., X_7)}{H(S_1, S_2, ..., S_{18})} \leq 8.0\%$$

The low information loss rate with EEDC can be explained as follows. Since the observations of any two sensor nodes within a single cluster are similar, their mutual information is large. That is, given the observation from one sensor node, the observations from other sensor nodes in the same cluster should not bring much extra information. On the other hand, since the observations of two sensor nodes from different clusters are not similar, their mutual information is small. In this case, the total information of the two sensor nodes should be close to the sum of their individual information, indicating that the joint entropy of the sensor nodes selected with EEDC from different clusters at any given time is approximately equal to the joint entropy of all sensor nodes.

3) Energy Saving: At any time instance, only one sensor node in a cluster is required to work. Since the extra working time of each sensor after its working shift, Δ_t, is far less than the one round of scheduling time, T, the energy cost during Δ_t can be safely ignored. In this case study, the eighteen sensor nodes were grouped into seven clusters with EEDC as shown in Figure 12. By calculating

$$\frac{2*2*5+4*4*2}{18} \approx 3,$$

we can see that without using EEDC, on average each sensor will spend three times more energy in sampling and data transmission.

D. Large-scale Synthetic Data Generation

Due to the high system cost, it is impractical for us to perform an experiment with hundreds of sensor nodes. In order to further investigate the performance of EEDC with large-scale networks, we synthetically generated large traces of spatially correlated dataset based on a mathematical model proposed in [11]. We utilized the software toolkit provided by [11] to extract the model parameters from small-scale real datasets and generate large-scale synthetic datasets based on the model parameters. The toolkit has been validated by comparing the statistical features of the synthetic dataset and the experimental dataset [11].

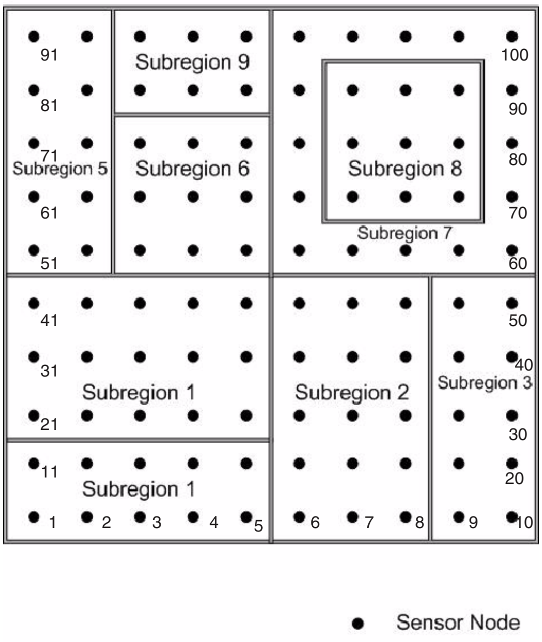

Fig. 13. The Field with Nine Distiguished Subregions

Initially, we used our test bed in Section V-A to collect a small-size real dataset. Then we utilized the synthetic data generation toolkit [11] on the dataset from each individual sub-region to generate a larger dataset for each individual sub-region. As the result, a field consisting of nine distinguished sub-regions with 100 sensor nodes in a 10×10 grid layout was generated, as shown in Figure 13.

E. Performance Results with Large-scale Synthetic Data

1) The Correctness of Clustering with EEDC: Since we know which sensor node belongs to which sub-region as a priori, it is easy to verify the correctness of the clustering algorithm. We set $m = 20, t = 95\%, gmax\_dst = 8$ distance units. The distant unit is defined as the distance between the two neighboring sensor nodes in a row. By calculating the pairwise dissimilarity measure and performing the clustering algorithm, we obtained nine clusters, each for a sub-region.

2) The Observation Fidelity with EEDC: Unlike the small-size real dataset, the large-scale synthetic dataset does not permit easy approximation of joint entropy. To avoid this problem, we use another performance metric, the difference distortion measure, which has been broadly used in image compression to evaluate the fidelity of a reconstructed image against the original image [17]. The difference distortion measure σ^2 is given by

$$\sigma^2 = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N} (X_{ij} - Y_{ij})^2}{M \times N}$$

where X_{ij} is the j-th actual sampling value from the i-th sensor node, Y_{ij} is the j-th sampling value of the i-th sensor node restored at the sink, N is the total number of sensor nodes, and M is the total number of samples from each sensor node. Note that since with EEDC, there is only one working node in each cluster at any given time, the sink node simply assumes that the sampling values of all sleeping sensor nodes in the same cluster are equal to that of the working node.

The absolute value of σ^2 is not meaningful without considering the degree of variation in magnitude. So we normalized the difference distortion measure by the average variation of samples and used it as the metric of observation fidelity. Formally,

$$\sigma^2_{norm} = \frac{\sigma^2}{\frac{\sum_{i=1}^{N} Var(X_i)}{N}},$$

where $Var(X_i)$ is the observation variation of the i-th sensor node.

By varying the value m in the magnitude m-similarity, we collected a set of performance data listed in Table I. Since with EEDC there is only one node working in a cluster at any time instance, the number of clusters is equal to the number of working nodes. Therefore, the number of clusters is proportional to the total number of transmitted data packets in the network, and hence it is an indicator of system energy consumption. From Table 1 we can see that with the decrease of the threshold m, the number of clusters and thus the energy consumption increase. But σ^2_{norm} decreases, indicating that the observation fidelity becomes better. This demonstrates clearly a trade-off between energy consumption and observation fidelity.

3) Energy Saving: At any time instance, only one sensor node in a cluster is required to work. Based on the clustering result, the 100 sensor nodes were grouped into 9 clusters with EEDC. By calculating

$$\frac{16*16+9*9*2+6*6+10*10*3+15*15*2}{100} \approx 12,$$

we can see that without using EEDC, on average each sensor will spend 12 times more energy in sampling and data transmission.

TABLE I

EEDC Performance Results

m	No. Of Cliques	σ^2_{norm}
20	9	0.0112
15	14	0.0074
10	23	0.0037

F. Performance Comparison with Coverage Based Scheduling

For comparison purpose, we also investigated the scheduling algorithm based purely on sensors' sensing coverage, that is, a sensor node is considered redundant if its sensing range is fully covered by its neighbors. We applied the coverage based scheduling algorithm to the above large-scale synthetic dataset, and compared the result to that obtained with EEDC.

Note that we did not compare EEDC to any specific coverage based scheduling algorithm. Instead, given a fixed sensing range, we calculated the optimal set for working sensor nodes. The optimization is in the sense that the size of the set of working sensor nodes is minimum, and thus the smallest energy is consumed to cover the whole area. Therefore, the number of working nodes is the lower bound that any coverage-guaranteed scheduling algorithm could achieve.

In the following experiment, we set sensors' sensing range as unit square, resulting in the minimum working sensor node set consisting of 25 evenly distributed working sensor nodes. The sink node assumes that the sampling value of a sleeping node is equal to that of the closest sensor node. For a specific sleeping node, if it has several closest working sensor nodes, the sink node picks up one working node randomly and takes its sampling value as the restored value of the sleeping node.

After calculation, the normalized distortion measure, σ^2_{norm}, was equal to 1.0452. Comparing to the results of EEDC in Table I, we can see that even if the number of data packets transmitted with EEDC is less than that with the coverage based sampling scheme, the observation fidelity of EEDC is much better. Experiments also show that changing the coverage range of each sensor node does not change this fact. The reason is straightforward: *scheduling based purely on sensors' coverage range may be inaccurate since it does not consider the features of the monitored phenomenon.* The above results clearly demonstrate the advantages of EEDC over coverage based scheduling schemes.

VI. Related Work

The problem of achieving energy efficiency in dense wireless sensor networks by decreasing spatial sampling rate has been explored to some extent. The basic idea of most existing approaches is to schedule sensor nodes to work alternatively to maintain a high coverage rate over the field of interests. A sensor node whose sensing range is fully covered by other working nodes is considered redundant and can be put into sleep. In this case, the geographical distance between sensor nodes is the only factor taken into consideration in sensor scheduling.

In [8], Hsin proposed a random scheduling scheme and a coordinated sleep scheduling scheme. In the random one, time is divided into slots with same length and each node determines randomly and independently at each slot whether it should be on or off. In the coordinated one, each node is assumed to be aware of its current location thus it can check whether it is totally redundant or not. If yes, it selects a random delay, picks up a sponsor node set, and broadcasts a request message to inform the nodes in the sponsor set to stay awake in a predefined sleep period. The sponsor node set consists of its neighbor nodes that can fully cover its sensing range. By adjusting the length of backoff time based on the relative residual energy, energy balance can be achieved.

A similar method was proposed in [18]. A coverage-based off-duty eligibility rule and a backoff-based node-scheduling scheme were adopted to guarantee a high sensing coverage.

In [20], a sensor node uses a probing mechanism to determine whether it should sleep. Once a sleeping node wakes up, it broadcasts a probing message to ask for reply from its neighboring active nodes. If no reply is received within a timeout period, the node assumes that there are no working nodes nearby and starts to work till it depletes its energy. Otherwise, it believes that it is redundant and goes to sleep again. The coverage rate can be changed by adjusting the probing range and the wakeup rate.

In [19], the authors divided the whole monitored field into grids and transformed the area coverage problem into the grid intersection point coverage problem. Each sensor node knows its location and its neighbors' locations. By exchanging messages with its neighbors through an adaptive energy-efficient sensing coverage protocol, each node is able to dynamically decide a schedule for itself, which guarantees the grid intersection points within its sensing range to be monitored by itself or by its neighbors at any time.

Rather than considering coverage as the only factor in scheduling, several pioneering methods have been proposed to adjust spatial sampling rate according to statistic features of the monitored phenomenon. In [6], a linear model was proposed to capture the spatial-temporal correlations among sampling data from different sources. With this model, most sensor nodes can be turned off and their readings can be estimated with certain accuracy by using the linear combinations of the data from working sensor nodes. However, in real world, a lot of systems may not be linear. Furthermore, the method of choosing the right working nodes has not been discussed in [6].

A novel approach to adjusting spatial sampling rate with the help of mobile sensor nodes was proposed in [2]. Mobility, combined with an adaptive algorithm, allows the system to get the most efficient spatial-temporal sampling rate to achieve a specified monitoring accuracy. Mobility can also make the system respond quickly to unpredictable environmental changes.

VII. Conclusion and On-going Work

In this paper, we design an Energy Efficient Data Collection (EEDC) framework that utilizes the spatial correlation to group sensor nodes into clusters. In the framework, the time-ordered sampling data from each sensor is treated as a time series and sensor nodes are separated based on the dissimilarity measure of the time series. With this method, the whole network is divided into several sub-regions, with each covered by a cluster of sensor nodes. Since the clusters are based on the features of sampling data, scheduling based on the clusters is much more accurate than scheduling based purely on the sensing range of sensor nodes. We also discuss the details of all major components in the EEDC framework, including the calculation of dissimilarity, sensor clustering, and sensor scheduling.

We thoroughly evaluate the performance of the EEDC framework with a real experiment based on MICA2 motes [5] and with a large-scale synthetic dataset. Experimental results demonstrate that the EEDC framework can effectively save energy without losing observation fidelity.

As our on-going work, we are investigating different approaches to measuring dissimilarity based on different application contexts. We hope to provide a set of criteria that can guide data analysis and simplify protocol design for specific applications.

Acknowledgement

This research was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and

Canada Foundation for Innovation (CFI). The authors would like to thank A. Jindal for his help on the generation of large-scale synthetic data and anonymous reviewers for their helpful comments.

REFERENCES

[1] I. F. Akyildiz, M. C. Vuran, and O. B. Akan, "On exploiting spatial and temporal correlation in wireless sensor networks," *Proceedings of WiOpt'04: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Cambridge, UK, March 2004.

[2] M. Batalin, G. Sukhatme, Y. Yu, M. Rahimi, G. Pottie, W. Kaiser, and D. Estrin, "Call and Response: Experiments in Sampling the Environment," *Proceedings of ACM SenSys 2004*, Baltimore, MD, Nov 2004.

[3] W. S. Conner, L. Krishnamurthy, and R. Want, "Making Everyday Life a Little Easier Using Dense Sensor Networks," *Proceedings of the ACM Ubicomp 2001*, Atlanta, GA, October 2001.

[4] T. M. Cover, and J. A. Thomas, *Elements of Information Theory,* John Wiley, 1991.

[5] Crossbow Technology Inc., http://www.xbow.com/Products/products.htm, Accessed in February 2005.

[6] F. Emekci, S. E. Tuna, D. Agrawal, and A. E. Abbadi, "BINOCULAR: A System Monitoring Framework," *Proceedings of First Workshop on Data Management for Sensor Networks (DMSN 2004)*, Toronto, Canada, August 2004.

[7] D. Ganesan, D. Estrin, and J. Heidemann, "Dimensions: Why do we need a new data handling architecture for sensor networks?" *Proceedings of the First Workshop on Hot Topics in Networks (Hotnets-I)*, Princeton, NJ, October, 2002.

[8] C. Hsin, and M. Liu, "Network Coverage Using Low Duty-Cycled Sensors: Random & Coordinated Sleep Algorithm," *Proceedings of Information Processing in Sensor Networks 2004*, Berkeley, CA, April 2004.

[9] J. Han, and M. Kamber, *Data Mining: Concepts and Techniques,* Morgan Kaufmann, 2001.

[10] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS 2002)*, Vienna, Austria, July 2002.

[11] A. Jindal and K. Psounis, "Modeling Spatially-correlated Sensor Network Data," *Proceedings of Sensor and Ad Hoc Communications and Networks 2004 (SECON2004)*, Santa Clara, CA, October 2004.

[12] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," *Proceedings of ICDCS Workshop on Distributed Event-based Systems (DEBS 2002),* Vienna, Austria, July 2002.

[13] C. Lund and M. Yannakakis, "On the hardness of approximating minimization problems," *Journal of the ACM 41*, 1994.

[14] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," *Proceedings of the WSNA 2002*, Atlanta, GA, September 2002.

[15] S. Pattem, B. Krishnamachari and R.Govindan, "The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks," *Proceedings of Information Processing in Sensor Networks 2004*, Berkeley, CA, April 2004.

[16] D. Petrovic, R. Shah, K. Ramchandran, and J. Rabaey, "Data funneling: routing with aggregation and compression for wireless sensor networks," *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA),* Anchorage, AK, May 2003.

[17] K. Sayood, *Introduction to Data Compression,* Morgan Kaufmann, 1996.

[18] D. Tian, and N.D. Georganas, "A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks," *Proceedings of ACM Workshop on Wireless Sensor Networks and Applications*, Atlanta, GA, October 2002.

[19] T. Yan, T. He, and J.A. Stankovic, "Differentiated Surveillance for Sensor Networks," *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, November 2003.

[20] F. Ye, G. Zhong, J. Cheng, S. Lu and L. Zhang, "PEAS: A Robust Energy Conserving Protocol for Long-lived Sensor Networks," *Proceedings of the 10th IEEE International Conference on Network Protocols*, Paris, France, November 2002.

[21] D. Zuckerman "NP-complete problems have a version that's hard to approximate," *Proceedings of the 8th IEEE Annual Structure in Complexity Theory Conference*, San Diego, CA, May 1993.