Continuous K-Means Monitoring with Low Reporting Cost in Sensor Networks

Ming Hua, Man Ki Lau, Jian Pei, *Senior Member, IEEE*, and Kui Wu, *Senior Member, IEEE*

Abstract—In this paper, we study an interesting problem: continuously monitoring k-means clustering of sensor readings in a large sensor network. Given a set of sensors whose readings evolve over time, we want to maintain the k-means of the readings continuously. The optimization goal is to reduce the reporting cost in the network, that is, let as few sensors as possible report their current readings to the data center in the course of maintenance.
To tackle the problem, we propose the reading reporting tree, a hierarchical data collection and analysis framework. Moreover, we develop several reporting cost effective methods using reading reporting trees in continuous k-means monitoring. First, a uniform sampling method using a reading reporting tree can achieve good quality approximation of k-means. Second, we propose a reporting threshold method which can guarantee the approximation quality. Last, we explore a lazy approach which can reduce the intermediate computation substantially. We conduct a systematic simulation evaluation using synthetic data sets to examine the characteristics of the proposed methods.

Index Terms—Sensor networks, clustering, k-means, low reporting cost

1 INTRODUCTION

Recently, more and more large wireless sensor networks have been used in many applications such as environment surveillance, manufacturing management, business asset administration, automation in transportation and health-care industry. Analyzing data collected from numerous sensors is one of the prominent issues in wireless sensor network applications. While a straightforward approach can collect data continuously from wireless sensor networks and conduct analysis in base stations, the power consumption of sensors is the major bottleneck of wireless sensor network lifetime. Often, once a wireless sensor node is deployed, it may be hard to recharge or replace its battery. Once the battery is used up, the sensor dies. With dead sensors, a wireless sensor network is handicapped. When many sensors die, the functionality of a sensor network degrades substantially.

The major power consumption for wireless sensors comes from sending out messages. Therefore, given a data analysis task, it is important to collect data from large wireless sensor networks such that only as few sensors as possible need to send out their readings while the data analysis quality is satisfactorily retained. This motivates the energy-preserving approaches for data collection and analysis on large sensor networks.

The latest sensor network techniques enable a sensor to sense multiple measures simultaneously. For example, an environmental surveillance sensor can detect temperature, humidity, and density of carbon dioxide at the same time. Therefore, more often than not, multidimensional data analysis such as clustering is needed for analyzing sensor network data.

K-means clustering [1], [2] is a popularly employed method in analyzing multidimensional data. Consider an l-dimensional space $D_1 \times \cdots \times D_l$. Let $dist(p, q)$ be the distance between two points p and q. Given a set of n points $S = \{s_1, \ldots, s_n\}$ in the space and a positive integer k, the *k-means problem* is to find k points (also known as centers) $c_1, \ldots, c_k$, which may or may not be in S, minimizing

$$\sum_{i=1}^{n} \min_{j=1}^{k} \{dist(s_i, c_j)\}.$$

In other words, each point is assigned to the closest center. The optimization objective is to minimize the sum of distances between the points and the closest centers.

Continuously monitoring k-means clustering has many important applications in data collection and analysis in large sensor networks.

As a concrete example, in underground structure monitoring, such as a coal mine monitoring system [3], we need to monitor not only the structure changes of underground tunnels, but also potential gas or water leaks. Equipped with gas sensors and accelerometers (sensors measuring acceleration and gravity induced reaction forces), a sensor node can report different measures simultaneously, which can be used to detect potential

Authors are in the alphabetical order.

- *M. Hua and J. Pei are with the School of Computing Science, Simon Fraser University, 8888 University Drive, Burnaby, BC Canada V5A 1S6. E-mail: {mhua, jpei}@cs.sfu.ca*
- *M. Lau is with MacDonald, Dettwiler and Associates Ltd., 13800 Commerce Parkway, Richmond, BC Canada V6V 2J3. E-mail: MALAU@mdacorporation.com*
- *K. Wu is with the Department of Computer Science, University of Victoria, PO Box 1700 STN CSC, Victoria, BC Canada V8W 3P6. E-mail: wkui@cs.uvic.ca*

Manuscript received December 6, 2007; revised ???? ??, 200?.

dangers such as collapses. To reduce false alarms in such a system, data correlation from different sensor nodes must be carefully investigated. K-means clustering comes into the right place as a simple yet effective approach for such a data analysis task. In this case, the typical status of sensors and the extreme situations such as collapsing, gas or water leaking are of interest. We can set the number of clusters to a not-too-small value such as 5 to 10 to capture the diversity of the distribution of sensor readings. By monitoring how the clusters change over time, we can monitor the distribution of the underground tunnel status. Rapid changes of cluster size and/or readings may indicate incidents that need human interaction. For example, in case of gas leaking or local collapsing, the sensors surrounding will form a distinguishing and rapidly growing cluster due to the fast changing values of gas density and/or acceleration. The clustering approach can effectively avoid false alarms since the changes of the cluster in size and collective readings are more robust.

To save energy, as suggested by the methods to be developed in this paper, a sensor can take a low frequency to report its readings when the readings do not change much, but adaptively more active in reading reporting when the readings evolve rapidly.

The problem of continuously maintaining k-means clustering with low reporting cost is a novel and challenging task though there are extensive studies on k-means clustering before. The existing work mainly improves the performance of k-means clustering from two aspects: reducing the number of scans and reducing the number of points needed to be checked. Those methods reducing the number of scans may not be applied to our problem since they still need to read all points at least once, which implies that all sensors need to report in our application example.

The methods using sampling to reduce the number of points accessed look promising. However, they do not address the issue of continuously maintaining the centers. Moreover, most of them are progressive: samples have to be drawn repeatedly until the quality guarantee is satisfied. In our application example, drawing samples repeatedly in a sensor network also incurs extra communication cost.

In this paper, we propose interesting and effective methods to tackle the problem. We make the following contributions.

First, we propose the reading reporting tree, a hierarchical data collection and analysis framework for continuous k-means monitoring for a large set of data points. The framework uses a conceptual tree to aggregate data points bottom up. We show that we can continuously monitor k-means effectively with a constant approximation ratio using a reading reporting tree. Moreover, the reporting cost can be reduced by a uniform sampling method.

Second, to further reduce the reporting cost, we observe that substantial changes of centers must be caused by substantial changes of some data points. Thus, to maintain k-means, we should pay more attention to those points whose values change substantially. We propose a reporting threshold method: only the points whose value changes are over a threshold should report. By setting the reporting threshold properly, we can guarantee the approximation quality of k-means.

Third, to further reduce the reporting cost even within the reading reporting tree, we explore a lazy method. In many situations a user does not want to update the k-means information if the centers do not change substantially. Accordingly, a point can report only if the change of its value may affect the centers substantially. Moreover, the bottom-up clustering analysis may terminate at an intermediate node of the reporting tree if the centers at a higher level are not affected substantially.

Last, we conduct a systematic simulation evaluation using synthetic data sets to examine the characteristics of the proposed methods.

Table 1 provides a cheat sheet of the notions used frequently in the paper.

TABLE 1
Frequently used notions.

Notion	Explanation
S	a set of points
s	a point in S
s^i	the value of s at instant i
h	the height of a reading reporting tree
t	the facility factor
$dsum()$	the sum of distances in a k-means clustering

The rest of the paper is organized as follows. In Section 2, we formally define the problem and briefly review the related work. In Section 3, we present a reading reporting tree framework for continuous k-means monitoring. A uniform sampling method is given in Section 4, and a reporting threshold method is developed in Section 5. In Section 6, we explore a lazy method. The simulation evaluation is reported in Section 7. Section 8 concludes the paper.

2 Problem Definition and Related Work

In this section, we define the problem formally and review the related work.

2.1 Problem Definition

We consider a set of points S. At a time instant i, the value of a point $s \in S$ is $s^i = (v_1, \ldots, v_l)$. In other words, a point in S can be regarded as a moving object in an l-dimensional space.

In this paper, 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 S. The points in S can be partitioned into

k exclusive subsets $S_1, \ldots, S_k$ according to their values at instant i: a point $s \in S$ is assigned to cluster S_i if

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

where $dist()$ is the distance function in question.

Definition 1 (K-means): Points $c_1, \ldots, c_k$ are the **k-means** of S if they minimize

$$\sum_{i=1}^{k} \sum_{s \in S_i} dist(s^i, c_i).$$

□

Some problems highly related to k-means are NP-hard, including the Minimum Sum-of-Squares Clustering (MSSC) problem and the Clustering to Minimize Sum of Diameters (CMSD) problem. For example, in the MSSC problem, a set of objects are partitioned into k clusters so that the sum of squared distances from the objects to the cluster mean is minimized. Formally, given a set of objects $X = \{X_1, \cdots, X_n\}$ where $X_i = (X_{i_1}, \cdots, X_{i_s}) \in R^S$ is a vector in Euclidean space R^S, and an integer $k \le n$, the MSSC problem is to find a partition of X into k disjoint subsets $C_1, \cdots, C_k$, such that $\sum_{i=1}^{k} \sum_{x \in C_i} \|x - \bar{x}_i\|^2$ is minimized, where $\|.\|$ denotes the Euclidean norm and $\bar{x}_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ is the mean of partition C_i ($1 \le i \le k$). The MSSC problem is proved NP-complete in [4].

As another example, the CMSD problem aims to partition the vertices of a complete graph with non-negative edge weights into k subsets, so that the sum of the diameter of the subgraph regarding each subset is minimized. Formally, given a complete graph $G(V, E)$ where V is a set of vertices and E is the set of edges in G, a non-negative weight $w(u, v)$ for each edge $(u, v) \in E$ where $u, v \in V$, and an integer $k \le |V|$, the CMSD problem is to partition V into k subsets $V_1, \cdots, V_k$ such that $\sum_{i=1}^{k} Diameter(V_i)$ is minimized, where $Diameter(V_i)$ is the largest weight of the edges in the complete subgraph of G induced on V_i. It is shown in [4] that when edge weights do not satisfy the triangle inequality, the CMSD problem is NP-complete for any fixed $k \ge 3$.

However, no proof of NP-hardness has been achieved to date [5] for k-means. On the other hand, no polynomial time algorithm has been found. Therefore, in this paper, we focus on approximation methods as many previous studies do.

We are interested in maintaining k centers over time. We assume that most of the time the value of a point evolves mildly over time. Abrupt big changes do happen to points, but with a low probability. This assumption is realistic for many applications such as surveillance sensor networks monitoring environment in forest and glaciers. This assumption is technically important since it heuristically allows us to use the clusters at the previous instants as the base for approximation to the new clusters at a later instant.

Our goal is to reduce the reporting cost of data points as much as possible, while the quality of the k-means information is retained.

Definition 2 (Reporting cost): The **reporting cost** at an instant t is the number of points whose current values are reported in order to update the k-means information.

□

Different from many previous studies on k-means where the cost measures focus on computation overhead, in this paper, our focus is on how often a point has to report its current value. Here, a point can determine whether it should report based on the changes of its value over time and the requests from the data center. In other words, we model an intelligent data collection unit such as a sensor as a point.

In this paper, we assume that a metric distance is used, where the triangle inequality holds.

2.2 Related Work

Our study is highly related to the existing work on k-means clustering on data streams from the clustering theory point of view. It is also related to the previous studies on clustering in sensor networks from the application point of view. We provide a brief review here and point out the differences.

2.2.1 *k-means Clustering on Data Streams*

Our study is related to data stream clustering. The values of points evolve over time. Thus, the values of one point over time can be modeled as a data stream.

A few studies have been conducted to investigate the problem of data stream clustering, such as [6], [7], [8], [9]. Different from our study where the number of points is fixed and the interest is on maintaining k-means of the current snapshot, most of the existing work on stream clustering assumes that new data points keep arriving. The task there is to maintain k-means of all data points seen so far or in a sliding window.

Nevertheless, some critical techniques in stream clustering can be borrowed to tackle the problem studied in this paper. Particularly, Guha *et al.* [7] developed a divide-and-conquer strategy, called Smaller Space, to cluster data streams. The Smaller Space method divides a data stream into chunks such that each chunk can be held in main memory. It clusters a chunk to obtain a set of cluster centers that are weighted by the number of points in the clusters. To obtain the k-means of the whole stream, it clusters the weighted centers in chunks. Although the nature of the data streams in [7] and the problem studied here are quite different, the idea of this strategy can be extended and applied to tackle the problem in this paper. We will discuss the details in the next section.

2.2.2 *Clustering in Sensor Networks*

Since we use data sensor networks as a motivating example of the problem studied here, our study is also related to clustering in sensor networks. Most of the previous studies on sensor network clustering focus on how to cluster sensors so that sensors having similar readings or behavior are grouped together. The main advantage of such clustering is that the sensor readings within a cluster may be similar and can be aggregated, so that the transmission cost can be reduced by limiting the number of outgoing messages (e.g., [10], [11], [12] and Reactive Sensor Networks (RSN) http://strange.arl.psu.edu/RSN/).

For example, Banerjee *et al.* [13] defined a sensor cluster as a set of connected sensors in a sensor network topology, with certain size constraints. Ideally, each node should only belong to one cluster. Thus, there is low overlapping among clusters. In order to find such clusters, the algorithm first derives a rooted spanning tree of the sensor network topology, and then partitions the spanning tree according to the clustering criteria.

To lower down the communication cost in sensor networks, Bandyopadhyay *et al.* [14] proposed a hierarchical clustering structure. A set of sensors are grouped together, and one of them becomes a *clusterhead*. In data collection, each sensor in the cluster sends its data to the clusterhead, and the clusterhead reports the aggregated data to the processing center. A distributed randomized algorithm was proposed to cluster the sensors. Each sensor takes a probability to become a clusterhead, and broadcasts itself to other sensors within certain hops. The sensors that are not clusterheads join the closest clusterhead. The optimal parameters of the clustering which minimize the communication cost are also derived. A similar data collection framework is also used in [15].

The above algorithm can be used to build a hierarchical structure in a sensor network to minimize the communication cost in data collection. 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 clusterheads, the lifetime of the whole cluster decreases. To tackle the problem, Younis *et al.* [16], [17] proposed *Hybrid Energy-Efficient Distributed clustering*, which periodically recomputes the clusterheads based on the residual energy of each sensor and its relationship to other sensors.

Meka *et al.* [18] defined a δ cluster as a set of sensors whose communication graph is connected and the distance of features between any pair of sensors in the cluster is at most δ. Finding δ clustering is proved to be NP-complete. An efficient distributed algorithm was proposed to compute high quality approximate clustering. In the hierarchical clustering structure of a sensor network, each cell represents a set of sensors. The sensor closest to the centroid of a cell is elected as the leader of the cell, and is called a *sentinel node*. The algorithm first picks the sentinel node at the root, and lets it expand to form a δ cluster. Then, the sentinel nodes at the lower levels grow to form δ clusters recursively. This process terminates when every node in the sensor network is included in a δ cluster.

In [19], a method is proposed to dynamically explore the spatial and temporal correlation of sensor readings, and cluster sensors accordingly for energy preserving data collection. The clustering algorithm continuously responds to spatial correlation changes and dynamically forms new clusters. The clustering criterion in [19], however, is different from the one in this paper, and the problem of k-means clustering has not been touched in [19].

Specifically, there are two key differences between our study and [19]. First, in this paper, we aim to cluster the sensor readings as the current values of points instead of sensor nodes in sensor networks. Second, we not only compute the initial clustering, but also monitor the clustering structure dynamically. Novel techniques are developed to improve the efficiency.

Prior to our study, Bash *et al.* [20] proposed an approximately uniform random sampling method for data collection in sensor networks. It tackles a problem different from ours in this paper: a spatial sample may result in a non-uniform sample of sensor nodes. To overcome the problem, the major idea is to use geographic routing, distributed computation of Voronoi regions and von Neumann's rejection method. Technically, the method utilizes the topology of the sensor network in question. Different from [20], the methods developed in this study do not rely on any network topology information.

Our study is also broadly related to the previous work on energy saving data collection [21], [22] and query evaluation [23], [24], [25], [26] in sensor networks. The major challenge is to collect the required data from sensor networks with high quality, and reduce the communication cost as much as possible. For example, [27] proposes an energy-efficient framework SAF to approximate query and cluster nodes in a sensor network. The major idea is to build models for readings of sensor nodes, and use the models to predict the readings. Moreover, nodes are clustered according to similarity. Different from the problem studied here, the clustering algorithm in SAF clusters sensor nodes according to the models of sensors stored in the data center.

3 Reading Reporting Trees

Definition 3 (Reading reporting tree): Given a set of points S, a **reading reporting tree**, as shown in Figure 1, is a tree satisfying the following requirements:

- The root node of the tree collects data and maintains the k-means information.
- Each data point is a leaf node in the tree.
- An internal node is called a *data collection node* if it is the parent of some leaf nodes. A data collection node has a fan-out of t^2k, where t is a *facility factor*.
- An internal node is called an *aggregation node* if it is not a data collection node. An aggregation node has a fan-out of t.

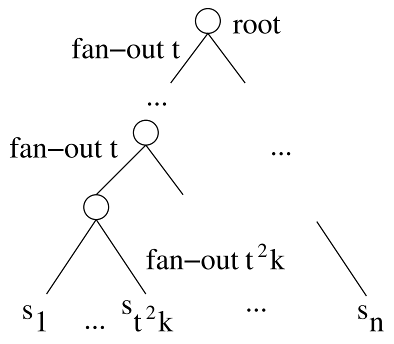


Fig. 1. A reading reporting tree.

- The tree is balanced. That is, the paths from the root node to all leaf nodes have the same length. □

Without loss of generality, we assume that the number of data points is $|S| = t^h k$, where $h > 0$ is the height of the reading reporting tree, that is, a path from the root to a leaf has h nodes inclusively. In other words, the reading reporting tree is full. In such a situation, $h = \log_t \frac{|S|}{k}$. When a reading reporting tree is not full, we can easily reduce the height of the tree by 1 and increase the fan-outs of the data collection nodes so that each data collection node has a fan-out of at least $t^2 k$. The major results in this paper still hold.

To compute the k-means information, straightforwardly, we can adopt the *hierarchical L-search method* [28]. The method works as follows.

1) Each point reports the current value to the parent node in the reading report tree, which is a data collection node.
2) A data collection node runs a k-means algorithm and clusters the $t^2 k$ points collected from its children into $t \cdot k$ centers. Each center carries a weight w which is the number of points assigned to it. Virtually, we treat the center as a set of w points at the identical location as the approximation of the cluster. The weights will be used in the next step. The data collection node reports the centers and the weights to the parent, which is an aggregation node.
3) An aggregation node runs a k-means algorithm and clusters the $t^2 k$ centers collected from its t children into $t \cdot k$ centers. When the k-means algorithm is applied, a center c of weight w from a child is treated as w points at the location of c. Again, each center generated in the aggregation node carries a weight which is the sum of the weights of the centers from the children nodes assigned to it. The aggregation node reports the centers computed and their weights to its parent.
4) The computation is conducted in a bottom-up way in the reading reporting tree. The root node generates k centers from the $t^2 k$ weighted centers received from its children.

Extending the ideas in [28], the above hierarchical L-search algorithm generates a good approximation of the k-means.

Theorem 1 (Quality – hierarchical L-search): The hierarchical L-search algorithm has an approximation factor of $\alpha_{L-search} = 2^{h-2} b^{h-1}$, where h is the height of the reading reporting tree, and b is the approximation factor of the k-means clustering algorithm used in the internal nodes (including the data collection nodes and the aggregation nodes) of the reading reporting tree. □

The proof of the theorem is provided in Appendix A.

The advantage of the hierarchical L-search method is that it can be used at anytime, and has a constant approximation factor (as long as a k-means method with a constant approximation factor is used in each node in the reading reporting tree). However, the disadvantage is that the reporting cost is high: every point has to report its current value.

Proposition 1 (Cost – hierarchical L-search): The hierarchical L-search algorithm has the reporting cost of $O(|S|)$, where S is the set of points. □

Can we reduce the reporting cost but still retain the good quality of k-means clustering? This is the topic of the rest of this paper.

4 A Uniform Sampling Method

Suppose the initial k-means information is obtained at the root node by running the hierarchical L-search method once. After a period, the values of some points may change. Now, let us consider how to update the k-means information.

To reduce reporting cost, instead of asking each point to report, an intuitive method is to derive the k-means information from a uniform sample of all points. With a large enough uniform sample, we can ensure the quality of the approximation with a high probability.

Technically, in the *uniform sampling method*, each point takes a probability of p to report. Then, the hierarchical L-search is run on the uniform sample. The k-means derived from the sample are used as the approximation of the k-means on all points.

Let O be a set of points and C be a set of points as the centers. The *sum of distances* of O using C is defined as

$$dsum(O, C) = \sum_{o \in O} \min_{c \in C} \{dist(o, c)\}$$

Let X be a uniform sample of the points in S. Let C_S be the centers of the exact k-means (that is, the optimal centers) on S, and C_X be the centers of the exact k-means on X. In the uniform sampling method, the hierarchical L-search method is used to approximate C_X, and the approximation of C_X is used as the approximation of C_S.

To measure the approximation quality, we assign the points in S into clusters using the centers in C_X. Then, the sum of distances using C_X is

$$dsum(S, C_X) = \sum_{o \in S} \min_{q \in C_X} \{dist(o, q)\}$$

The sum of distances of the k-means is

$$dsum(S, C_S) = \sum_{o \in S} \min_{q \in C_S} \{dist(o, q)\}$$

We have the following result.

Theorem 2 (Uniform sampling method): For any δ $(0 < \delta < 1)$ and ϵ $(0 < \epsilon < 1)$, in the uniform sampling method, if the sample size

$$|X| \geq \frac{3 \ln \frac{2}{\delta}}{dsum(S, C_S)\epsilon^2} |S|$$

then

$$dsum(S, C_X) \leq \frac{1+\epsilon}{1-\epsilon} dsum(S, C_S) \quad (1)$$

with a probability at least $(1-\delta)$. Moreover, the uniform sampling method has a constant approximation factor with a high probability with respect to the hierarchical L-search method. □

The proof of Theorem 2 is given in Appendix B.

Theorem 2 shows that using a reading reporting tree, a uniform sampling method can achieve good quality provided that the sample size is large enough. Apparently, the sampling rate in the uniform sampling method should be at least $\frac{3 \ln \frac{2}{\delta}}{dsum(S, C_S)\epsilon^2}$. Thus, the reporting cost of the uniform sampling method is as follows.

Proposition 2: In the uniform sampling method, the expected number of points reporting at an instant is

$$Cost_{uniform-samp} \geq \frac{3 \ln \frac{2}{\delta}}{dsum(S, C_S)\epsilon^2} |S|$$

where S is the set of data points. □

One drawback of the uniform sampling method is that, as shown in Theorem 2, the required sample size depends on $dsum(S, C_S)$, which is unknown to users. Although a loose upper bound of $dsum(S, C_S)$ can be easily obtained by randomly choosing k points in the data space as the centers and calculating the sum of distances of the points to the centers, such an estimation is ineffective in practice. In implementation, we can choose a sample size based on the k-means estimation at the last instant due to the assumption that the data changes are often mild.

5 A Reporting Threshold Method

The uniform sampling method treats each point the same: each point reports with the same probability. However, if the values of many points may evolve mildly, their values may be relatively stable and thus may have little effect on the changes of the centers. Can we take the advantage of this relative stability to reduce reporting cost?

Here, we propose a simple *reporting threshold method* which works as follows. Let $\Delta > 0$ be a *change threshold*. Suppose at instant i_0, a point s reports the current value s^{i_0} (to its parent). At an instant $i > i_0$ such that s does not report at any instant between i_0 and i, s reports again if and only if $dist(s^i, s^{i_0}) \geq \Delta$, where s^i is the value of s at instant i. In other words, a point reports again only if its value changes at least Δ from the value it reports last time. As the initialization step, we run the hierarchical L-search method at instant 1 such that every point reports.

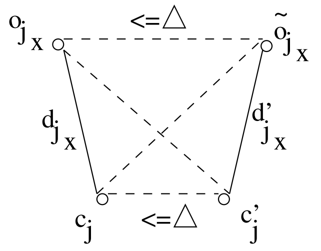


Fig. 2. The relation between d_{j_x} and d'_{j_x}.

At a data collection node, for each point s, a last reported value $\tilde{o}$ is maintained. Apparently, $dist(o, \tilde{o}) < \Delta$, where o is the current value of s. Otherwise, the point should update its value. The reporting threshold method uses $\tilde{o}$ to calculate the k-means as the approximation of the current k-means. Like the hierarchical L-search method, the data collection and clustering procedure runs bottom-up in the reading reporting tree.

Now, let us examine the clustering quality of the reporting threshold method. For the set of points $S = \{s_1, \ldots, s_n\}$, let o_i and $\tilde{o_i}$ $(1 \leq i \leq n)$ be the real value of point s_i and the value reported for clustering, respectively. The reporting threshold method ensures $dist(o_i, \tilde{o_i}) < \Delta$.

Lemma 1: Let ds be the sum of distances of the exact k-means on $o_1, \ldots, o_n$ and $\widetilde{ds}$ be the sum of distances of the exact k-means on $\tilde{o_1}, \ldots, \tilde{o_n}$. Then, $\widetilde{ds} < ds + 2n\Delta$, where Δ is the reporting threshold.

Proof: Let $c_1, \ldots, c_k$ be the centers in the exact k-means on $o_1, \ldots, o_n$, and $\tilde{c_1}, \ldots, \tilde{c_k}$ be the centers in the exact k-means on $\tilde{o_1}, \ldots, \tilde{o_n}$.

For each center c_j, let $o_{j_1}, \ldots, o_{j_l}$ be the values assigned to c_j. We can calculate the corresponding center c'_j of $\tilde{o_{j_1}}, \ldots, \tilde{o_{j_l}}$. That is, we form clusters of $\tilde{o_1}, \ldots, \tilde{o_n}$ synchronizing with $o_1, \ldots, o_n$ in partitioning. Since $dist(o_i, \tilde{o_i}) < \Delta$, $dist(c_j, c'_j) < \Delta$.

Let $d_{j_x} = dist(o_{j_x}, c_j)$ and $d'_{j_x} = dist(\tilde{o_{j_x}}, c'_j)$. As illustrated in Figure 2, $|d_{j_x} - d'_{j_x}| < 2\Delta$.

Let $ds' = \sum_{i=1}^{n} d'_i$. Then, $ds' \leq ds + 2n\Delta$. Clearly, $ds' \geq \widetilde{ds}$ since $\widetilde{ds}$ is the optimum on $\tilde{c_1}, \ldots, \tilde{c_k}$. Thus, we have $\widetilde{ds} < ds' \leq ds + 2n\Delta$. The lemma is proved. □

Theorem 3 (Quality–reporting threshold): The reporting threshold method has an approximation factor of $2^{h-2}b^{h-1}$ to the optimum k-means solution, where h is the height of the reading reporting tree, and b is the approximation factor of the k-means clustering algorithm used in the internal nodes of the reading reporting tree.

Proof: According to Theorem 1, the hierarchical L-search method introduces an approximation factor $2^{h-2}b^{h-1}$. The reporting threshold method uses the hierarchical L-search method to approximate the optimum k-means solution on the values reported. Lemma 1 shows that the exact k-means on the stored values approximate the exact k-means on the real current values by an absolute error bound $2n\Delta$. $2n\Delta$ is a constant and does not affect the approximation factor of the reporting threshold method. Thus, we have the theorem. □

The above theoretical analysis shows that the reporting threshold method only introduces an error bounded by a constant (when the set of points and the threshold are fixed). Moreover, we can control the error bound by setting the reporting threshold.

How much can be saved in reporting cost by the reporting threshold method?

Theorem 4 (Cost–reporting threshold): Let S be the set of points. Under the assumptions that (1) the values of points are independent from each other; and (2) the changes of values of points follow a normal distribution with a mean of μ and a standard deviation of σ, in the reporting threshold method, the expected number of points reporting at an instant is

$$(1 - erf(\frac{\Delta}{\sigma\sqrt{2}}))|S|$$

where

$$erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt = \frac{2}{\sqrt{\pi}}\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n+1}}{n!(2n+1)}$$

is the Gauss error function, and Δ is the reporting threshold.

Proof: Consider a point $s \in S$. Suppose $\tilde{o}$ is the value of s last reported. At the current instant, s has the current value o. It reports again if and only if $dist(o, \tilde{o}) \geq \Delta$.

Since the changes of the values follow a normal distribution, the probability $Pr(dist(o, \tilde{o}) < \Delta)$ is the area under the curve of the probability density function of normal distribution so that $\mu - \Delta < x < \mu + \Delta$. That is, $Pr(dist(o, \tilde{o}) < \Delta) = erf(\frac{\Delta}{\sigma\sqrt{2}})$.

Thus, $Pr(dist(o, \tilde{o}) \geq \Delta) = 1 - Pr(dist(o, \tilde{o}) < \Delta) = 1 - erf(\frac{\Delta}{\sigma\sqrt{2}})$. There are $|S|$ points in total. The expectation stated in the theorem follows. □

One possible drawback of the reporting threshold method is that it may be very sensitive to the threshold. The data from sensors can be very noisy. Compared to the uniform sampling approach, the negative effect of noise can be enlarged in the reporting threshold method if the threshold is not set properly.

6 A Lazy Method

All the methods we discussed so far are aggressive in data collection and clustering: they conduct clustering at every instant. Moreover, a k-means clustering procedure is run at every internal node of the reading reporting tree. In the situations where the values of points evolve slowly, the k-means clusters change slowly and are relatively stable. Instead of being aggressive, can we be adaptive to the changes so that more reporting cost can be saved even for the internal nodes?

Here we propose a lazy method. The central idea is that, at each internal node of the reading reporting tree, the k-means centers in a previous instant are reused as much as possible unless the changes are large enough.

Technically, for each internal node u in the reading reporting tree, at an instant, u reports the new centers to its parent only if using the new centers results in a change of at least τ in the sum of distances than using the centers u reports last time, where τ is the change threshold on the sum of distances. We describe the details in two cases.

In the first case, let us consider a data collection node u. Suppose $s_1, \ldots, s_{t^2k}$ are the children of u. Then, u remembers the centers $c_1, \ldots, c_k$ and the sum of distances ds that it reports to the parent last time, as well as the weights of the centers $w_1, \ldots, w_k$, that is, w_i ($1 \leq i \leq k$) points are assigned to center c_i in the k-means partitioning.

When u receives the new values from the children, u runs the k-means procedure to calculate the new centers and the sum of distances ds_{new}. u also tries to greedily assign the new readings to the old centers $c_1, \ldots, c_k$ as follows. For each point o of the new readings, o is assigned to the nearest center whose capacity is not full. We assign points to centers in the distance ascending order. A center c_i is full once it is assigned w_i points. The sum of distances between the points to the assigned centers ds'_{new} is calculated. u reports the new centers to its parent only if $|ds_{new} - ds'_{new}| \geq \tau$.

Lemma 2: ds'_{new} can be computed in time $O(t^2k^2(k + \log t^2k))$.

Proof: Each data collection node has t^2k children. Sorting the leaf nodes in the distance ascending order takes time $O(t^2k \log t^2k)$. When some centers are full, those leaf nodes using some full center as the nearest center need to update the distance to the nearest center, and be inserted into the right position in the sorted list. For such a leaf node, the cost is $O(k + \log t^2k)$. Such adjustments can happen at most $O(t^2k \cdot k)$ times. Thus, the overall complexity is $O(t^2k \log t^2k + t^2k \cdot k \cdot (k + \log t^2k)) = O(t^2k^2(k + \log t^2k))$. □

In the second case, let us consider an aggregation node u. The children of u report to u the centers in the children nodes with the corresponding weights. We only need to treat each updated center c with weight w from a child as w points at location c. The method in the first case can be applied straightforwardly.

We can also use the observation in Lemma 1 to save more. Let u be a data collection node, which has t^k children. If every child of u has a change of value less than $\frac{\tau}{t^k}$, then the change of the sum of distances must be also less than τ. Clearly, no children of u need to report, and u does not need to run the k-means clustering procedure.

Using the above observation, at an instant, a point informs its parent if its value changes at least $\frac{\tau}{t^2k}$. Then, the parent informs its children to update the values if it receives the change notification from at least one child. Once the new values arrive, a k-means clustering is conducted and if the change in the sum of distances is at least τ, an update is reported to the parent.

We claim that the lazy method has a good approximation quality as follows.

Theorem 5 (Quality – lazy method): The lazy method has an approximation factor of $2^{h-2}b^{h-1}$ to exact k-means, where h is the height of the reading reporting tree, and b is the approximation factor of the k-means clustering algorithm used in the internal nodes of the reading reporting tree.

Proof: At each internal node in the reading reporting tree, when the node reports, the approximation factor b is satisfied. When the node does not report, that is, the lazy method uses threshold τ to save communication, we have $\widetilde{ds} \leq b \cdot ds + \tau$ where ds is the optimum sum of distances and $\widetilde{ds}$ is the approximation using the centers of some previous instant. In other words, the approximation factor b still holds. According to Theorem 1, the approximation factor $2^{h-2}b^{h-1}$ is achieved at the root of the reading reporting tree. □

The lazy method can be viewed as pushing the reporting threshold method into every internal node.

Theorem 6 (Cost – lazy method): Let S be the set of points. Under the assumptions that (1) the values of points are independent from each other; and (2) the changes of values of points follow a normal distribution with a mean of μ and a standard deviation of σ, in the lazy method, the expected number of points reporting at an instant is

$$\frac{|S|}{t^2k}(erf(\frac{\tau}{t^2k\sigma\sqrt{2}}))^{t^2k}$$

where $erf(x)$ is the Gauss error function, and τ is the change threshold on the sum of distances.

Proof: Using the same idea in the proof of Theorem 4, we can show that $erf(\frac{\tau}{t^2k\sigma\sqrt{2}})$ is the probability that a point does not raise a change notification to its parent. A point does not report if itself and all its siblings do not raise a change notification. The probability is $(erf(\frac{\tau}{t^2k\sigma\sqrt{2}}))^{t^2k}$. The points are partitioned into $\frac{|S|}{t^2k}$ groups at the leaf node level. Thus we have the formula in the theorem. □

The lazy method can be effective if the values of the points are relatively stable, such as surveillance sensors in forest. In such a situation, the lazy method can reduce the cost of clustering procedures and communication. It is change-driven – an internal node is triggered only if the points that it manages detect some significant changes.

7 SIMULATION EVALUATION

Table 2 summarizes the quality guarantees and the expected reporting cost in those methods. In this section, we empirically evaluate the methods proposed in this paper. Particularly, we test the algorithms in the aspects of clustering quality, reporting cost, and scalability.

All the experiments were conducted on a PC computer with a 3.0 GHz Pentium 4 CPU, 1.0 GB main memory, and a 160 GB hard disk, running the Microsoft Windows XP Professional Edition operating system, Our algorithms were implemented in Microsoft Visual Studio 2005.

7.1 Simulation Setup

There are some existing synthetic data generation methods in sensor networks in literature. Yu *et al.* [29], [30] developed a synthetic data generation framework for sensor networks, for the purpose of statistics estimation of data, data compression, and data estimation. Jindal and Psounis [31], [32] provided methods to generate data in sensor networks with various degrees of spatial correlation. Other synthetic data generation methods include [29] and [33].

The existing methods are not suitable for our simulation purpose, since the generated data may not reflect the characteristics of clusters in sensor readings and the temporal evolving behavior of clusters. Therefore, we generate synthetic data sets to simulate the cluster-evolving scenarios in sensor networks as follows[1].

A data set contains n points in an l-dimensional space $D_1 \times \cdots \times D_l$. The domain of each dimension is $[0, 2000]$. By default, $n = 1,280$ and $l = 2$.

At instant 1, 98% of the points in the data set form k (by default, $k = 5$) clusters with equal size and the rest 2% of points are noise points that do not belong to any cluster. The number of points in each cluster is $\lceil \frac{98\%n}{k} \rceil$. The cluster centers are uniformly distributed in space $D_1 \times \cdots \times D_l$. The cluster radius follows the normal distribution $N(20, 1)$. For each point s in a cluster with center c and radius r, the reading of s in dimension D_j ($1 \leq j \leq l$) follows the normal distribution $N(c_j^1, \sigma)$ in range $[c_j^1 - r, c_j^1 + r]$, where c_j^1 is the reading of center c in dimension D_j at instant 1 and $\sigma = N(20, 1)$. The noise points are uniformly distributed in space $D_1 \times \cdots \times D_l$.

The sensor readings at instant i ($i > 1$) are generated to simulate the following cluster-evolving scenarios.

LM (local move): for each point, the current reading deviates slightly from the reading of the same point at the previous instant. Particularly, at instant i ($i > 1$), for each point s, the reading of s in dimension D_j ($1 \leq j \leq l$) is generated following the normal distribution $N(s_j^{i-1}, 10)$, where s_j^{i-1} is the reading of s in dimension D_j at instant $i - 1$.

RS (radius shrinking): at the current instant, the radius of each cluster shrinks compared to the radius of the same cluster at the previous instant. Particularly, for each point s, let s_j^{i-1} be the reading of s in dimension D_j ($1 \leq j \leq l$) at instant $i - 1$ ($i > 1$). The distance between s and the cluster center c is $|s_j^{i-1} - c_j^{i-1}|$, where c_j^{i-1} is the reading of center c in dimension D_j at instant $i - 1$. At instant i, the reading of center c does not change (that is, $c_j^i = c_j^{i-1}$), while the reading of s in dimension D_j is

1. The data sets are available at http://www.cs.sfu.ca/~jpei/Software/SESER.rar

TABLE 2
The summary of the quality guarantees and expected reporting cost of the methods.

Method	Approximation factor	Reporting cost
Hierarchical L-search	$2^{h-2}b^{h-1}$	$O(\|S\|)$
Uniform sampling	$2^{h-2}b^{h-1}\frac{1+\epsilon}{1-\epsilon}$	$O(\frac{3\ln\frac{2}{\delta}}{dsum(S,C_S)\epsilon^2}\|S\|)$
Reporting threshold	$2^{h-2}b^{h-1}$	$O((1-erf(\frac{\Delta}{\sigma\sqrt{2}}))\|S\|)$
Lazy	$2^{h-2}b^{h-1}$	$O(\frac{\|S\|}{t^2k}(erf(\frac{\tau}{t^2k\sigma\sqrt{2}}))^{t^2k})$

changed to $s_j^i = c_j^i + (s_j^{i-1} - c_j^i)\alpha$, where $\alpha \in (0,1)$ is the radius changing ratio. By default, $\alpha = 0.6$.

RE (radius expanding): at the current instant, the radius of each cluster enlarges compared to the radius of the same cluster at the previous instant. The reading of each point at instant i is generated in the same way as the situation of radius shrinking. The only difference is that the radius changing ratio α is greater than 1. By default, $\alpha = 2$.

RB (radius bumping): the radius of each cluster changes from the previous instant, and some clusters merge into each other. The data generation for the radius bumping scenario for instant i is the same as that in the situation of radius expanding. The only difference is that the radius changing ratio α is large so that some clusters may merge. By default, $\alpha = 6$.

CC (center change): the center of each cluster moves. Particularly, at instant i $(i > 1)$, the reading of a center c in dimension D_j $(1 \leq j \leq l)$ is generated following the normal distribution $N(c_j^{i-1}, 100)$, where c_j^{i-1} is the reading of c in dimension D_j at instant $i-1$. For each point s in the cluster with center c, the reading of s in dimension D_j at instant i is changed to $s_j^i = c_j^i + (s_j^{i-1} - c_j^{i-1})$, where s_j^{i-1} is the reading of s in dimension D_j at instant $i-1$.

ME (membership exchange): many points leave the clusters they belong to at the previous instant and join a new cluster at the current instant. Particularly, at instant i, we randomly select a cluster C as the dissolving cluster. Each point s in C joins another cluster C' that is randomly selected from the rest clusters. The reading of s at instant i is generated in the same way as the existing member points in C'.

We test the approximation quality of each algorithm in various cluster evolving situations. The height of the reading report tree is set to 4, and the facility factor $t = 4$. The number of clusters is 5.

So far there does not exist a polynomial time algorithm to compute the optimal k-means. Thus, we use the popularly adopted randomized algorithm to compute an approximation of the optimal results. That is, as the baseline method, we run the k-means algorithm offline on the whole data set, and obtain the sum of distance as the benchmark. The clustering results in the proposed methods are compared with the benchmark values. Previous studies have shown that the offline method can often achieve very good approximation to the optimal results, and has a theoretically provable approximation ratio 2 [34].

The error rate of each algorithm is defined as

$$Error\ rate = \frac{\widehat{dsum} - dsum}{dsum \cdot k}$$

where $dsum$ is the sum of distances computed by the offline k-means algorithm and $\widehat{dsum}$ is the sum of distances computed by an approximation algorithm proposed in this paper. k is the number of clusters.

7.2 Clustering Quality

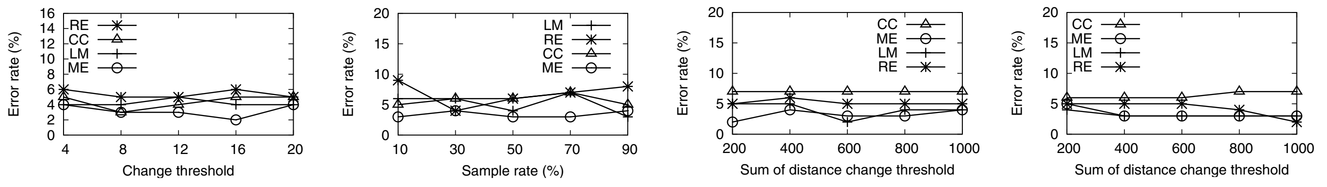
In this subsection, we report the results on testing the clustering quality of our methods. Since the k-means algorithms are randomized algorithms in nature, we run each of our experiments 5 times, and report the median of the results. Choosing the median is to avoid the ill-effect of some extreme outlier values.

We compare four methods: the reporting threshold method (RT), the uniform sampling method (US), the lazy reporting threshold method (lazyRT) and the lazy uniform sampling method (lazyUS). The lazy uniform sampling method is to combine the lazy method with the uniform sampling: if k-means clustering has to be conducted in an internal node in a reading reporting tree, a uniform sample is draw from its children to derive the clustering information.

7.2.1 Clustering Quality of the Four Algorithms

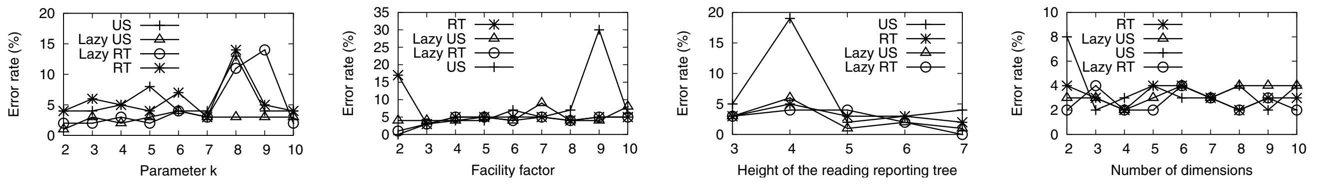
The error rates of the four methods in various situations of cluster changes are reported in Figure 3. Since the trends of the error rates in the situations of radius shrinking (RS), radius expanding (RE), and radius bumping (RB) are similar, we only show the curves of the radius expanding (RE) situation as the representative and omit the curves of (RB) and (RS) to make the figures more legible.

Figure 3(a) reports the error rates of the reporting threshold method. Generally, as the change threshold increases, fewer points will report their values, and thus the error rates are expected to increase. However, in the situations of local move (LM) and membership exchange (ME), the error rates drop slightly when the change threshold is greater than 12 (there are very few new point values reported, and the clustering is mainly based on the previous data). The reason is that, the sum of distances is stable in those two situations, therefore, computing the clustering using most of the previous



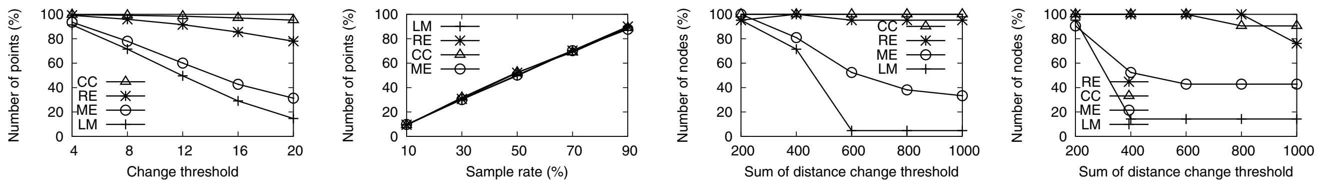
(a) Error rate of RT. (b) Error rate of US. (c) Error rate of lazy RT. (d) Error rate of lazy US.

Fig. 3. Approximation quality in different situations of cluster changes.



(a) Error rate vs. k. (b) Error rate vs. facility factor. (c) Error rate vs. height (d) Error rate vs. dimensionality.

Fig. 4. Approximation quality with respect to parameters of the reading reporting tree and the dimensionality.



(a) Efficiency of RT. (b) Efficiency of US. (c) Efficiency of lazy RT. (d) Efficiency of lazy US.

Fig. 5. Reporting cost in different clustering evolving situations.

readings actually gives a good approximation of the real clustering.

Figure 3(b) shows that the error rates of the uniform sampling method are stable in general and decrease slightly as the sample rate increases. This is because, more point values are collected when the sample rate is higher. In general, the error rates of the uniform sampling method are higher than the error rates of the reporting threshold method. The reason is, the reporting threshold method uses both the previous readings of those stable points and the updated current values of those significantly changed points. Thus, more accurate information is captured in the reporting threshold method.

Figures 3(c) and 3(d) show the error rates of the lazy reporting threshold method and the lazy uniform sampling method, respectively. In general, when the sum of distances change threshold becomes larger, fewer internal nodes report their changes, so the error rates become higher. This trend is observed in the experiments, though the increase of error rates is quite mild. The error rates of the lazy methods are all below 10%, which illustrate their effectiveness.

In summary, the four methods all provide good approximation quality of the real clusterings in various cluster evolving situations. The reporting threshold method is more accurate than the uniform threshold method since the reporting threshold method reuses the information at the previous instant. The lazy methods can provide high quality approximation in most clustering evolving situations, except for the center change case.

7.2.2 Effects of the Parameters of Reading Reporting Trees on Quality

The four algorithms proposed in this paper use the reading reporting tree framework. A reading reporting tree takes three structural parameters: the number of clusters k, the facility factor t, and the height of the tree h. Moreover, it is well known that clustering is sensitive to the dimensionality of data. In our case, the dimensionality is the number of measures detected by a point[2].

We tested the clustering quality with respect to the four parameters listed above. Here, we use the situation of local move to report the results. The results on the other situations of cluster evolving are similar.

We set the parameters for the algorithms as follows. In the reporting threshold method, we set the change threshold to 10. In the uniform sampling method, we set sampling rate to 50%. In the two lazy methods, we set the change threshold to 600.

2. In the applications of sensor networks, typically, this is a small positive integer (e.g., about 3-5) in most of the state-of-the-art sensor networks.

Figure 4(a) shows the error rates of the four algorithms with respect to different number of clusters k. It is interesting that as k increases from 2 to 8, the error rates also increase; but when k further increases from 9 to 10, the error rates drop down. The reason is, when k is very small, each cluster contains a large set of points, and is relatively stable. When the number of clusters increases, the clusters become smaller and are easier to be affected by the change of data. However, when the number of clusters is large, most of the clusters are local. Local changes in data cannot affect most clusters.

Figure 4(b) shows the error rates of the algorithms with respect to different facility factors. We can observe the a similar trend as in Figure 4(a). This is because, when t is very small, there are very few point values in each micro-cluster, therefore, the cluster centers reported in the internal nodes are very close to the actual point values. The approximation is accurate. As t increases, the error rates also go up. This is because, the current values of points are approximated by cluster centers in the internal nodes of a reading report tree, which leads to an accuracy loss. However, when $t \geq 7$, the error rates start to decrease, because when there are enough points in each micro-cluster, the increasing number of cluster centers led by a larger value of t becomes a good approximation of other readings in the same cluster.

In Figure 4(c), the error rates decrease when the height of the reading reporting tree increases. This is simply because the number of points increases dramatically when the height of the reporting becomes larger. With more data, the clustering quality can be expected better. Technically, the sum of distances in the actual clustering also increases dramatically. This sum is used as the denominator of the error rate calculation. In other words, the approximation made by our methods are proportionally stable with respect to the sum of distances in the actual clustering. The results also translate to a highly desirable conclusion: our methods have lower error rates (that is, better approximation quality) on larger data sets.

Figure 4(d) shows the error rates with respect to dimensionality up to 10. The error rates of our methods are stable in general, though some exceptional points do exist. The results clearly show that our methods can handle the dimensionality high enough for a few applications such as sensor networks detecting multiple measures.

7.3 Reporting Cost

The major purpose of our four methods is to save reporting cost. In this subsection, we test the effect of reporting cost reduction of all the four algorithms with respect to different cluster evolving situations and structural parameters of the reading reporting tree.

7.3.1 reporting cost in the Four Algorithms

We evaluated the reporting cost of the four algorithms in terms of the number of points that report their current values during the clustering process. The results are shown in Figure 5.

Figure 5(a) shows the number of points reported with respect to change threshold in the reporting threshold method. As expected, when the change threshold increases, fewer points will report their values. Among the different cluster evolving situations, most points report their values in the center change situation, since all the point values change substantially in order to make the cluster center change. In the local move situation, the fewest points report their values, because a point does not report its current value if the change between the previous reading and the current reading is within the change threshold. When the change threshold is 20, very few points report their current values, but the error rates are still lower than 10%, as shown in Figure 3(a), which shows the effectiveness of the reporting threshold method.

Figure 5(b) shows that the number of points reported increases linearly as the sample rates increases. When the sample rate is 10%, there are fewer than 200 points reported, but the error rates are still lower than 10%, as shown in Figure 3(b). If we compare the error rates of the reporting threshold method and the uniform sampling method, it is interesting to observe that when the number of samples reported is similar in the two methods, the error rates of the reporting threshold methods are lower than the error rates of the uniform sampling method, which again verifies the effectiveness of the reporting threshold method.

In Figures 5(c) and 5(d), we plot the percentage of number of internal nodes reported in the lazy reporting threshold method and the lazy uniform sampling method. As the sum of distances change threshold increases, fewer internal nodes report their centers. It is clear that in the lazy methods, only a small portion of internal nodes report their new centers, which verifies the effectiveness of the lazy methods in reducing the reporting cost.

In summary, the number of points reported in the clustering can be controlled by different parameters in the four algorithms. Reducing the number of reporting points can reduce the reporting cost. Moreover, our experimental results show that the approximation quality is not affected significantly while the number of reporting points can be reduced substantially. This property clearly shows the effectiveness of our methods.

8 Conclusions

In this paper, we tackled a novel and interesting problem: continuously monitoring k-means with low reporting cost. We proposed a reading reporting tree structure and developed a set of methods. Our methods reduce the number of points that need to report, and thus save the reporting cost.

As future work, it is interesting to exploit our methods in applications like data collection in sensor networks.

In such applications, in addition to reporting cost, there are other types of cost which have not been included in our model. For example, sensing different measures may have different cost. Integrating those types of cost in a comprehensive model is interesting and challenging.

Acknowledgement

We sincerely thank the anonymous reviewers and Professor Diane J. Cook for their constructive suggestions which help to improve the quality of the paper.

The research of Ming Hua, Man Ki Lau and Jian Pei was supported in part by an NSERC Discovery Grant and an NSERC Discovery Accelerator Supplements Grant. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies. The research of Kui Wu was supported in part by an NSERC discovery grant and a CFI new opportunity grant.

References

[1] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, pp. 129–137, 1982.
[2] J. B. Macqueen, "Some methods of classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathemtical Statistics and Probability*, 1967, pp. 281–297.
[3] M. Li and Y. Liu, "Underground structure monitoring with wireless sensor networks," in *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2007, pp. 69–78.
[4] P. Brucker, "On the complexity of clustering problems," *Lecture Notes in Economics and Mathematical Systems*, vol. 157, pp. 45–54, 1978.
[5] S. Durocher, *Geometric Facility Location under Continuous Motion*, ser. Ph.D. thesis. University of British Columbia, April 2006.
[6] C. C. Aggarwal, J. Han, J. Wang, and P. Yu, "A framework for clustering evolving data streams," in *Proc.the 19th Int. Conf. on Very Large Data Bases (VLDB'03)*, Berlin, Germany, September 2003.
[7] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 515–528, 2003.
[8] L. O'Callaghan, A. Meyerson, R. Motwani, N. Mishra, and S. Guha, "Streaming-data algorithms for high-quality clustering," in *ICDE '02: Proceedings of the 18th International Conference on Data Engineering*. Washington, DC, USA: IEEE Computer Society, 2002, p. 685.
[9] J. Beringer and E. Hüllermeier, "Online clustering of parallel data streams," *Data Knowl. Eng.*, vol. 58, no. 2, pp. 180–204, 2006.
[10] M. Younis, M. Youssef, and K. Arisha, "Energy-aware routing in cluster-based sensor networks," in *MASCOTS '02: Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02)*. Washington, DC, USA: IEEE Computer Society, 2002, p. 129.
[11] S. Ghiasi, "Optimal energy aware clustering in sensor network," *Sensor*, vol. 2, pp. 258–269, 2002.
[12] S. Yoon and C. Shahabi, "The clustered aggregation (cag) technique leveraging spatial and temporal correlations in wireless sensor networks," *ACM Trans. Sensor Networks*, vol. 3, no. 1, p. 3, 2007.
[13] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks," in *INFOCOM*, 2001, pp. 1028–1037.
[14] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *INFOCOM*, 2003.
[15] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660–670, 2002.
[16] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in *INFOCOM*, 2004.
[17] ——, "Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. Mob. Comput.*, vol. 3, no. 4, pp. 366–379, 2004.
[18] A. Meka and A. K. Singh, "Distributed spatial clustering in sensor networks," in *EDBT*, 2006, pp. 980–1000.
[19] C. Liu, K. Wu, and J. Pei, "An energy-efficient data collection framework for wireless sensor networks by exploiting spatiotemporal correlation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 7, pp. 1010–1023, 2007.
[20] B. A. Bash, J. W. Byers, and J. Considine, "Approximately uniform random sampling in sensor networks," in *Proceedings of the 1st international workshop on Data management for sensor networks (DMSN'04)*. New York, NY, USA: ACM, 2004, pp. 32–39.
[21] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *ICDE*, 2006, p. 48.
[22] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *VLDB*, 2004, pp. 588–599.
[23] D. Q. Goldin, "Faster in-network evaluation of spatial aggregationin sensor networks," in *ICDE*, 2006, p. 148.
[24] A. Manjhi, S. Nath, and P. B. Gibbons, "Tributaries and deltas: Efficient and robust aggregation in sensor network streams," in *SIGMOD Conference*, 2005, pp. 287–298.
[25] X. Yang, H.-B. Lim, M. T. Özsu, and K.-L. Tan, "In-network execution of monitoring queries in sensor networks," in *SIGMOD Conference*, 2007, pp. 521–532.
[26] A. Bhattacharya, A. Meka, and A. K. Singh, "Mist: Distributed indexing and querying in sensor networks using statistical models," in *VLDB*, 2007, pp. 854–865.
[27] D. Tulone and S. Madden, "An energy-efficient querying framework in sensor networks for detecting node similarities," in *Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems (MSWiM'06)*. New York, NY, USA: ACM, 2006, pp. 191–300.
[28] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams," in *Proc. IEEE Symposium on Foundations of Computer Science (FOCS'00)*, Redondo Beach, CA, 2000, pp. 359–366.
[29] Y. Yu, D. Ganesan, L. Girod, D. Estrin, and R. Govindan, "Synthetic data generation to support irregular sampling in sensor networks," 2003.
[30] Y. Yu, D. Estrin, M. Rahimi, and R. Govindan, "Using more realistic data models to evaluate sensor network data processing algorithms," in *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 569–570.
[31] A. Jindal and K. Psounis, "Modeling spatially-correlated data of sensor networks with irregular topologies," in *IEEE SECON 2005: Proceedings of the Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*. IEEE Computer Society, 2005, pp. 305–316.
[32] ——, "Modeling spatially correlated data in sensor networks," *ACM Trans. Sen. Netw.*, vol. 2, no. 4, pp. 466–499, 2006.
[33] Y.-A. L. Borgne, M. Moussaid, and G. Bontempi, "Simulation architecture for data processing algorithms in wireless sensor networks," in *AINA '06: Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 2 (AINA'06)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 383–387.
[34] V. V. Vazirani, *Approximation Algorithms*. Springer, March 2004.
[35] D. Angluin and L. G. Valiant, "Fast probabilistic algorithms for hamiltonian circuits and matchings," in *Proceedings of the ninth annual ACM symposium on Theory of computing (STOC'77)*. New York, NY, USA: ACM Press, 1977, pp. 30–41.

Appendix A
Proof of Theorem 1

To prove Theorem 1, we need the following two lemmas.

Lemma 3: Given a set of points S, let $dsum_{opt}(S)$ be the sum of distances in the optimum k-means on S. Consider an arbitrary partitioning of S into l exclusive subsets $S_1, \ldots, S_l$. Let $dsum_{opt}(S_i)$ be the sum of distances in the optimum k-means on S_i $(1 \leq i \leq l)$. Then,

$$\sum_{i=1}^{l} dsum_{opt}(S_i) \leq dsum_{opt}(S).$$

Moreover, let S' be the set of weighted centers of $S_1, \ldots, S_l$, that is, each point $c \in S'$ is a center in some S_i carrying a weight w where w is the number of points in S_i assigned to c. Let $dsum_{opt}(S')$ be the sum of distances in the optimum k means on S', then

$$dsum_{opt}(S') \leq dsum_{opt}(S) + \sum_{i=1}^{l} dsum_{opt}(S_i).$$

Proof: Let C_S be the set of optimum k centers on S. Consider subset $S_i \subseteq S$. Let $dsum_{C_S}(S_i)$ be the sum of distances of assigning points in S_i to centers C_S. Clearly, $dsum_{opt}(S_i) \leq dsum_{C_S}(S_i)$. Otherwise, $dsum_{opt}(S_i)$ is not the optimum. Therefore,

$$\sum_{i=1}^{l} dsum_{opt}(S_i) \leq \sum_{i=1}^{l} dsum_{C_S}(S_i) = dsum_{opt}(S).$$

The first part is proved.

Consider a point $p \in S'$ with weight w_p. Let $c_{opt}(S', p)$ be the center to which p is assigned in the optimum k-means on S'. The sum of distances in the optimum k-means on S' is

$$\sum_{p \in S'} dist(p, c_{opt}(S', p)) \cdot w_p.$$

Since each weighted point p is a weighted center in $S_1, \ldots, S_l$, for any point $s \in S$ there exists a point $p_s \in S'$ such that s is assigned to p_s. Thus, the sum of distances in the optimum k-means on S' can also be written as

$$\sum_{s \in S} dist(p_s, c_{opt}(S', p_s)).$$

Let $c_{opt}(S, s)$ be the center to which s is assigned in the optimum k-means on S, and p_s is the center where s is assigned in some S_i. According to the triangle inequality, we have $dist(c_{opt}(S, s), p_s) \leq dist(s, p_s) + dist(s, c_{opt}(S, s))$.

Since

$$\sum_{s \in S} dist(s, p_s) = \sum_{i=1}^{l} dsum_{opt}(S_i)$$

and

$$\sum_{p} dist(p, c_{opt}(S, p)) = dsum_{opt}(S),$$

we have the second inequality in the lemma. □

Lemma 4: In the hierarchical L-search method using a reading reporting tree, each aggregation node in the reading reporting tree which is a grand parent of a leaf node finds $O(k)$-centers with an approximation factor of $2b^2$, where b is the approximation factor for the k-means clustering algorithm used in the internal nodes of the reading reporting tree.

Proof: According to the definition, each aggregation node which is a grand parent of a leaf node has t children, and each of its child collects t^2k points. Let X_i be the set of data points collected by the i-th child of an aggregation node o.

Using the first item in Lemma 3, we know

$$\sum_{i=1}^{t} dsum_{opt}(X_i) \leq dsum_{opt}(\cup_{i=1}^{t} X_i).$$

Let $dsum_b(X_i)$ be the sum of distances in the k-means solution on X_i found by a b-approximation algorithm. Then,

$$\sum_{i=1}^{t} dsum_b(X_i) \leq b \cdot dsum_{opt}(\cup_{i=1}^{t} X_i).$$

Let X' be the set of weighted centers at node o. Using the second item in Lemma 3, we know

$$dsum_{opt}(X') \leq dsum_{opt}(\cup_{i=1}^{t} X_i) + \sum_{i=1}^{l} dsum_{opt}(X_i).$$

Then,

$$\begin{aligned} dsum_b(X') &\leq b(dsum_{opt}(\cup_{i=1}^{t} X_i) + \textstyle\sum_{i=1}^{l} dsum_{opt}(X_i)) \\ &\leq b(dsum_{opt}(\cup_{i=1}^{t} X_i) + dsum_{opt}(\cup_{i=1}^{t} X_i)) \\ &= 2b dsum_{opt}(\cup_{i=1}^{t} X_i) \end{aligned}$$

Since the clustering result of each aggregation node is obtained by first clustering the data in its children and then clustering the weighted centers to $O(k)$ centers, the approximation factor is $2b \cdot b = 2b^2$. □

Proof of Theorem 1: Let the approximation factor for the clustering result at an aggregation node at the j-th level is A_j. The path from such a node to a leaf node has length j.

By the assumption that a b-approximation algorithm is used in each internal node, we know that $A_1 = b$. From Lemma 4, we know that the approximation factor follows a recurrence $A_j = A_{j-1} 2b$ $(j \geq 2)$. Solving the recurrence, we have $A_j = 2^{j-1} b^j$. Since the root node is at level $h - 1$, the approximation factor at the root node is $2^{h-2} b^{h-1}$. □

Appendix B
Proof of Theorem 2

To prove Theorem 2, we need the following lemma.

Lemma 5: If the sample size $|X| \geq \frac{3 \ln \frac{2}{\delta}}{dsum(S, C_S)\epsilon^2} |S|$, then

$$Pr\{|dsum(X, C_S) - E[dsum(X, C_S)]| > \epsilon E[dsum(X, C_S)]\} \leq \delta \quad (2)$$

and

$$Pr\{|dsum(X, C_X) - E[dsum(X, C_X)]| > \epsilon E[dsum(X, C_X)]\} \leq \delta \quad (3)$$

where $E[Y]$ is the expectation of variable Y.

Proof: Following with Chernoff-Hoeffding bound [35], we have

$$Pr\{|dsum(X,C_S) - E[dsum(X,C_S)]| > \epsilon E[dsum(X,C_S)]\} \leq 2e^{-\frac{\epsilon^2 E[dsum(X,C_S)]}{3}} \leq \delta \quad (4)$$

Since X is a uniform sample of S, we have

$$E[dsum(X,C_S)] = \frac{|X|}{|S|}dsum(S,C_S)$$

Inequality 4 can be rewritten as

$$Pr\{|dsum(X,C_S) - E[dsum(X,C_S)]| > \epsilon E[dsum(X,C_S)]\} \leq 2e^{-\frac{\epsilon^2 \frac{|X|}{|S|} dsum(S,C_S)}{3}} \leq \delta$$

Thus, if $|X| \geq \frac{3\ln\frac{2}{\delta}}{dsum(S,C_S)\epsilon^2}|S|$, Equation 2 holds.

Similarly, following with Chernoff-Hoeffding bound [35], we have

$$Pr\{|dsum(X,C_X) - E[dsum(X,C_X)]| > \epsilon E[dsum(X,C_X)]\} \leq 2e^{-\frac{\epsilon^2 \frac{|X|}{|S|} dsum(S,C_X)}{3}} \leq \delta$$

Thus, Equation 3 holds if $|X| \geq \frac{3\ln\frac{2}{\delta}}{dsum(S,C_X)\epsilon^2}|S|$.

Clearly, we have $dsum(S,C_X) \geq dsum(S,C_S)$. So,

$$\frac{3\ln\frac{2}{\delta}}{dsum(S,C_X)\epsilon^2}|S| \leq \frac{3\ln\frac{2}{\delta}}{dsum(S,C_S)\epsilon^2}|S|$$

Therefore, if $|X| > \frac{3\ln\frac{2}{\delta}}{dsum(S,C_S)\epsilon^2}|S|$, Equation 3 holds immediately. □

Proof of Theorem 2: Since C_X is the set of optimal centers in sample X, we have $dsum(X,C_X) \leq dsum(X,C_S)$. Apparently,

$$E[dsum(X,C_S)] = \frac{|X|}{|S|}dsum(S,C_S)$$

and

$$E[dsum(X,C_X)] = \frac{|X|}{|S|}dsum(S,C_X)$$

Using Lemma 5, we have, when $|X| \geq \frac{3\ln\frac{2}{\delta}}{D(S,C_S)\epsilon^2}|S|$,

$$dsum(X,C_S) \leq (1+\epsilon)E[dsum(X,C_S)] = (1+\epsilon)\frac{|X|}{|S|}dsum(S,C_S)$$

and

$$dsum(X,C_X) \geq (1-\epsilon)\frac{|X|}{|S|}dsum(S,C_X)$$

with a probability higher than $(1-\delta)$. Thus, we have

$$(1+\epsilon)\frac{|X|}{|S|}dsum(S,C_S) \geq (1-\epsilon)\frac{|X|}{|S|}dsum(S,C_X)$$

Inequality 1 follows with the inequality immediately.

The uniform sampling method uses the hierarchical L-search method to approximate $dsum(S,C_X)$. Theorem 1 indicates that the approximation factor is $2^{h-2}b^{h-1}$ where b is the approximation factor of the k-means algorithm used in each internal node of the reading reporting tree, and h is the height of the tree. Therefore, the approximation factor of the uniform sampling method is $2^{h-2}b^{h-1}\frac{1+\epsilon}{1-\epsilon}$ with a high probability $(1-\delta)$ when the sample size is large enough. The theorem is proved. □

Ming Hua received her B.Sc. degree in Computer Science from Fudan University, China, in 2004. She is currently a Ph.D. candidate in School of Computing Science at Simon Fraser University, Canada. Her research interests lie in analyzing and mining uncertain data and its applications.

Man Ki Lau received her B.Sc. and M.Sc. degrees in Computing Science from Simon Fraser University in 2005 and 2007, respectively. Her research interests include trustworthy database indexing, data mining on large wireless sensor networks and the related applications. She is currently a software engineer in MacDonald, Dettwiler and Associates Ltd., working on embedded database systems.

Jian Pei received his Ph.D. degree in Computing Science from Simon Fraser University, Canada, in 2002. He is currently an Associate Professor of Computing Science and the director of Collaborative Research and Industrial Relations in School of Computing Science at Simon Fraser University, Canada. His research interests can be summarized as developing effective and efficient data analysis techniques for novel data intensive applications. Currently, he is interested in advanced techniques of data mining, data warehousing, online analytical processing, database systems, and information retrieval, as well as their applications in web search, sensor networks, health-informatics, bioinformatics, and business. His research has been supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), the National Science Foundation (NSF) of the United States, Microsoft, IBM, Hewlett-Packard Company (HP), Business Objects, the Canadian Imperial Bank of Commerce (CIBC), and the SFU Community Trust Endowment Fund. He has published prolifically in refereed journals, conferences, and workshops. He is an associate editor of IEEE Transactions on Knowledge and Data Engineering, and Intelligent Data Analysis. He has served regularly in the organization committees and the program committees of many international conferences and workshops, and has also been a reviewer for the leading academic journals in his fields. He is a senior member of the Association for Computing Machinery (ACM) and the Institute of Electrical and Electronics Engineers (IEEE). He is the recipient of the British Columbia Innovation Council 2005 Young Innovator Award, an IBM Faculty Award (2006), the KDD'08 Best Application Paper Award, and an IEEE Outstanding Paper Award (2007).

Kui Wu (M'02-SM'07) received the Ph.D. degree in computing science from the University of Alberta, Canada in 2002. He then joined the Department of Computer Science, University of Victoria, Canada, where he is currently an associate professor. His research interests include mobile and wireless networks, sensor networks, network performance modeling and evaluation, and network security. He is a senior member of IEEE and a member of ACM.