Outlier Detection Based Fault Tolerant Data Aggregation for Wireless Sensor Networks

Suat Ozdemir and Yang Xiao†

Computer Engineering Department, Gazi University, Ankara, TURKEY TR-06570
†Department of Computer Science, The University of Alabama Tuscaloosa, AL 35487-0290
suatozdemir@gazi.edu.tr, †yangxiao@cs.ua.edu

Abstract—Data aggregation protocols are essential for wireless sensor networks to prolong network lifetime by reducing energy consumption of sensor nodes. For mission critical wireless sensor networks, however, not only the energy consumption of sensor nodes but also the correctness of the data aggregation results is critical. This paper presents a fault tolerant data aggregation scheme that eliminates the false data sent by malfunctioning and/or compromised sensor nodes. To conserve energy while eliminating false data, an in-network outlier detection technique that is based on Locality Sensitive Hashing (LSH) scheme is used. The simulation results show that the proposed scheme is able to reduce the number of false data transmissions thereby increasing the data aggregation accuracy.

I. INTRODUCTION

Recent advances in wireless communications accelerated the deployment of Wireless Sensor Networks (WSNs) that typically consist of a large number of small, low-cost sensor nodes distributed over a large area with a powerful sink node that collects and analyzes readings of sensor nodes. Sensor nodes rely on small batteries and usually capable of measuring physical phenomenons such as temperature, sound, vibration, pressure, etc. In many cases, WSNs are employed to gather data from a hostile or unattended area which makes sensor node battery replacement too expensive or even impossible [1]. Hence, a WSN must perform the data gathering task in an energy-efficient manner so that its lifetime is prolonged. Data aggregation is implemented in wireless sensor networks to eliminate data redundancy, reduce data transmission, and improve data accuracy. It is shown that data aggregation results in better bandwidth and battery utilization [2], [3], which enhances the network lifetime because communication constitutes 70% of the total energy consumption of the network [4]. In WSNs, data aggregation is performed by sensor nodes, called data aggregators. Data aggregators are not only responsible from collecting and summarizing data but also in-network analysis of the collected data and trigger alarms based on this analysis [1].

In addition to energy efficient data gathering requirement, majority of WSN applications require real-time data mining of sensor data to promptly make intelligent decisions [6], hence identifying outliers is an important challenge for monitoring, fault diagnosis and intrusion detection in WSNs. In data mining domain, outliers are “events with extremely small probability of occurrence” [7]. In WSN domain, however, outliers are defined as “measurements that significantly deviate from the normal pattern of sensed data”. The difference between these two outlier definitions is due to unique properties of WSNs, such as cheap hardware or exhausted batteries. make them especially prone to outliers. Such properties of WSNs lead to generation of false/faulty sensor data. False data negatively influence the quality of aggregated data which is used for in-network decision making process. Since WSNs are usually employed to monitor the physical world phenomenons such as forest fire or earthquake, a phenomenon that is not accurately detected may be catastrophic. It is clear from the above discussion that outlier detection mechanisms must be implemented in WSNs so that data aggregators, i.e., decision makers, can correctly trigger alarms. However, outlier detection process is a memory and communication consuming task by its nature [10]. In distributed and resource constrained environments, such as WSNs, identifying outliers without increasing the communication overhead is a challenging task. Moreover, sensor nodes suffer from the severely limited memory capabilities. Therefore, in WSNs, in-network outlier detection approaches that reduce communication and memory consumption of sensor nodes must be employed.

In this paper, we propose a Fault Tolerant Data Aggregation scheme using an in-network outlier detection mechanism, called FTDA. The outlier detection mechanism is based on the Locality Sensitive Hashing (LSH) technique [8]. The LSH algorithm used in FTDA allows compact representation of sensor data which reduces the communication overhead of outlier detection. FTDA takes advantage of LSH technique by estimating the similarity of sensor data from their compact sketches (LSH codes). The data aggregator perfors data aggregation based on these LSH codes.

Our contribution in this paper is twofold. First, we propose a novel fault tolerant data aggregation scheme, FTDA, using an in-network outlier detection mechanism based on LSH technique. With the help of LSH technique, FTDA protocol is able to detect outliers in a distributed and energy efficient manner. Second, using LSH codes, FTDA protocol eliminates the redundant data transmission from sensor nodes to data aggregators thereby incrementing the efficiency of data aggregation process. The rest of the paper is organized as
follows. In Section II, the related work in data aggregation and outlier detection in WSN domain is presented. Section III explains the system model. Section IV explains the proposed protocol in detail. Performance analysis and simulation results are presented in Section V. Finally, concluding remarks are made in Section VI.

II. RELATED WORK

In wireless sensor network domain, secure data aggregation problem is studied extensively for years [3]. Recently, outlier detection in WSNs is became another attractive research area for researchers. The authors of [9] introduce a framework for cleaning and querying noisy sensors. The authors present an in-network Bayesian approach to reduce the uncertainty of the data due to random noise. To obtain a better estimation of the sensor node readings, the authors combine the prior knowledge of the real sensor reading, the noise characteristics of the sensor node, and the observed noisy reading. The authors propose several algorithms based on the introduced uncertainty models and evaluate the proposed algorithms. A comprehensive survey of outlier detection techniques is presented in [10]. To detect outliers in WSNs, the authors of [11] investigate the augmentation of sensor network queries by statistical models. The authors argue that a statistical model may offer a more reliable way to gain insight into the physical phenomena observed. Using statistical models, the authors propose an approach to detect outliers in streaming sensor data. The authors of [12] propose a histogram-based method to detect outliers in a communication efficient manner. A declarative data cleaning mechanism over sensor node data streams is introduced in [13]. A fuzzy logic based approach is proposed in [14] to infer the correlation among measurements from different sensors. The proposed technique assigns a confidence value to each measurement and then performs an aggregated weighted average scheme. The authors of [15] propose a technique based on a weighted moving average to estimate actual sensor readings.

III. SYSTEM MODEL AND PRELIMINARIES

We consider a large sensor network with densely deployed sensor nodes that are assigned unique identification numbers. Due to the dense deployment, sensor nodes have overlapping sensing regions and events are detected by multiple sensor nodes, thereby requiring data aggregation to reduce amount of data transmission. Sensor nodes have limited computation and communication capabilities whereas the base station is assumed to have no computation and communication constraints. The network is divided into clusters and each cluster has a dynamically selected data aggregator node. The details of cluster forming and data aggregator selection processes are out of scope of this paper. Data are periodically collected and aggregated in data aggregation sessions.

A. Outlier Definition

Defining outliers according to the latest reading of sensor nodes is shown to be a simple but unreliable technique [7]. FTDA detects outliers based on the last $m$ data readings of sensor nodes. To define outliers, let us first show how to measure the similarity between data of two sensor nodes. Let $v_i$ be the set of the latest $m$ readings collected by sensor node $S_i$. Also let $\Theta$ be a similarity threshold for a similarity metric $\theta$ where $\theta : R^m \rightarrow [0,1]$. Sensor nodes $S_i$ and $S_j$ are similar if $\theta(v_i,v_j) > \Theta$. Based on this similarity definition, we define local outliers in a cluster as follows.

**Definition 1. (Local Outlier)** Assume that $S_i$ and $S_j$ belong to same cluster. Sensor node $S_i$ is a local outlier if there are less than $\text{minSup}_{\text{local}}$ sensor node $S_j$ that satisfies $\theta(v_i,v_j) > \Theta$ where $\text{minSup}_{\text{local}}$ is minimum support value for the cluster.

It is worth to note that a sensor node may be labeled as a local outlier due to an event occurred in the neighboring cluster. For example, consider the fire monitoring scenario given in Figure 1 where cluster A, B, and C form a neighboring cluster group. When a fire started inside cluster A, it is expected that the sensor nodes of cluster B that are located close to cluster A detect the fire as well. Temperature readings of such nodes deviate significantly from the temperature readings of the other sensor nodes in cluster B. As a result, data aggregator of cluster B labels these nodes as local outliers. Hence, this process is not sufficient to label a sensor node as outlier. Hence, the data aggregator determines the outliers after communicating with its neighboring data aggregators.

![Fig. 1. Sensor nodes may be affected by the events occurred in the neighboring clusters. Cluster A, B, and C form a neighboring cluster group.](image)

**Definition 2. (Outlier)** Assume that $S_i$ and $S_j$ belong to a neighboring cluster group. Sensor node $S_i$ is an outlier if there are less than $\text{minSup}_{\text{group}}$ sensor nodes $S_j$ that satisfy $\theta(v_i,v_j) > \Theta$ where $\text{minSup}_{\text{group}}$ is minimum support value for the neighboring cluster group.

In FTDA, minimum support values $\text{minSup}_{\text{local}}$ and $\text{minSup}_{\text{group}}$ are dynamic system parameters that depend on the application and the collected data type. Based on the network size and local node density, the base station dynamically may alter minimum support values for different regions of the network. This approach reduces the errors due to usage of single minimum support value for outlier detection [16].

B. Distance and Similarity Metrics

To be able to measure the similarity between sensor node data sets, we need a distance metric. Let $P$ denote a set of data points and assume that $P$ has cardinality $n$. The points $p$ from $P$ belong to a $d$-dimensional space $\mathbb{R}^d$ and $p_i$ denotes $i$th coordinate of $p$, for $i = 1 \ldots d$. In $P$ the distance between
any pair \( p \) and \( q \), is defined as
\[
||p - q||_s = \left( \sum_{i=1}^{d} |p_i - q_i|^s \right)^{1/s}
\]
where \( s > 0 \). In general, \( s \) is taken as 2 and the distance is called Euclidean Distance. In this paper, we use Euclidean distance to compute the distance between sensor node data sets.

FTDA is independent from the similarity metric, hence any metric such as the cosine similarity, the correlation coefficient or the Jaccard coefficient can be used. FTDA employs cosine similarity which can be defined as follows:
\[
\cos(\theta(v_i, v_j)) = \frac{v_i \cdot v_j}{||v_i|| \cdot ||v_j||}
\]
where \( v_i \) and \( v_j \) denote data vectors of sensor nodes \( S_i \) and \( S_j \).

C. Locality Sensitive Hashing

Consider \( p \) and \( q \) which are some data points in \( \mathbb{R}^d \). If the distance between \( p \) and \( q \) is less than \( R \) then \( p \) is an \( R \)-near neighbor of \( q \). Basically, the LSH algorithm outputs if there is a \( R \)-near neighbor for a data point or not, hence the LSH algorithm relies on the existence of locality-sensitive hash functions. Let \( \mathcal{H} \) be a family of hash functions mapping \( \mathbb{R}^d \) to some universe \( Y \). Let’s assume that there is a function \( h \) in \( \mathcal{H} \). Furthermore, for points \( p \) and \( q \) \( h(p) = h(q) \). Under these assumptions, the family \( \mathcal{H} \) is called locality sensitive if it satisfies the following condition.

**Definition 3. (Locality Sensitiveness)** A hash function family \( \mathcal{H} \) is called locality sensitive if for any two points \( p, q \in \mathbb{R}^d \),
- if \( ||p - q|| \leq R \) then \( \Pr_{\mathcal{H}}[h(q) = h(p)] \geq P_1 \),
- if \( ||p - q|| \geq cR \) then \( \Pr_{\mathcal{H}}[h(q) = h(p)] \leq P_2 \),

where \( c \) is a constant, \( P_1 = 1 - \frac{2}{e} \), and \( P_2 = 1 - \frac{2}{eR} \). An LSH family must satisfy \( P_1 > P_2 \) [17]. This LSH family can determine that if two data points are in the \( R \)-near neighborhood of each other. In order to use LSH technique on sensor node data sets, we need an LSH algorithm that can work on vectors. In [8], a random hyperplane based LSH algorithm for vectors is proposed. For vectors \( u, v \in \mathbb{R}^d \), let us consider the cosine similarity metric that is the angle between the two vectors, \( \theta(u, v) = \arccos \left( \frac{uv}{||u|| ||v||} \right) \) and define the hash function \( h_r \) as
\[
h_r(u) = \begin{cases} 
1 & \text{if } r \times u \geq 0; \\
0 & \text{if } r \times u < 0.
\end{cases}
\]
then for vectors \( u \) and \( v \)
\[
\Pr[h_r(u) = h_r(v)] = 1 - \frac{\theta(u, v)}{\pi} \tag{3}
\]
\[
\theta(u, v) = \pi \times (1 - Pr) \tag{4}
\]
This random hyperplane based hash function can measure the similarity between any pair of sets. However, the hash function measures the similarity in terms of the angle between two vectors. Computation of an angle between two vectors is not a trivial task for a resource constrained sensor node. Therefore, following the method described in [18], we can rewrite the above equation in terms of hamming distance.
\[
D_h(LSH_u, LSH_v) = b \times (1 - Pr) \tag{5}
\]
where \( LSH_u, LSH_v \in [0, 1]^b \) are the LSH codes of vectors \( u \) and \( v \), respectively and \( D_h(LSH_u, LSH_v) \) is the hamming distance between \( LSH_u \) and \( LSH_v \). Each LSH code is length of \( b \)-bit which is much smaller than original vectors (i.e., data sets) \( u \) and \( v \). Using Equation 4, we can rewrite the above equation as follows:
\[
D_h(LSH_u, LSH_v) = b \times \frac{\theta(u, v)}{\pi} \tag{6}
\]
The above formula enables sensor nodes to measure the similarity of their data sets by simple bit comparisons. However, now we need to express the similarity threshold \( \Theta \) in terms of hamming distance as well. Using the Equation 6 we can write the similarity threshold as
\[
\Theta_{D_h} = b \times \frac{\Theta}{\pi} \tag{7}
\]
The next section explains how sensor nodes use \( D_h(LSH_u, LSH_v) \) and \( \Theta_{D_h} \) to detect outliers and redundant data.

IV. FTDA Protocol

FTDA protocol consists of three phases, namely (i) data collection and LSH code generation, (ii) outlier detection and redundant data elimination, and (iii) data aggregation. These three phases are periodically realized in each cluster. In what follows, we explain each phase in detail.

A. Phase 1. Data Collection and LSH Code Generation

In FTDA, data collection and aggregation is performed in sessions. Data aggregators inform their cluster members at the beginning of each data collection phase. In each data collection session, each sensor node senses the environment \( m \) times and stores the sensed values. Assuming that each sensed value is \( n \) bits, each sensor node has a data vector of size \( (m \times n) \)-bit. Sending this \( (m \times n) \)-bit data to the data aggregator results in rapid exhaustion of a sensor node’s battery. In order to reduce the amount of data transmission, sensor nodes generate LSH codes of their data vectors. As shown in the previous section, LSH codes can represent sensor data using less number of bits. A sensor node applies LSH algorithm to its data and obtain a \( b \)-bit LSH code where \( b < (m \times n) \). It is necessary to note that there is a tradeoff between the values of \( (m \times n) \) and \( b \) in terms of outlier detection probability. When \( (m \times n) \) and \( b \) values are close to each other, the outlier detection ability of the protocol increases. Using Equation 2, we can compute the probability \( P \) that LSH codes of data vectors \( u \) and \( v \) are equal. Hence, the probability of a successful similarity test can be expressed by the following cumulative function of a binomial distribution:
\[
P_{\text{similar}} = \sum_{i=0}^{\Theta_{D_h}} \binom{b}{i} P^i (1 - P)^{b - i} \tag{8}
\]
B. Phase 2. Outlier Detection and Redundant Data Elimination

Each data aggregator requests sensor nodes in its cluster to send their LSH codes for the current data aggregation session. Sensor nodes send their LSH codes along with their unique sensor node IDs. Using Equation 6 and 7, the data aggregator compares the LSH codes of any sensor node pair. The data aggregator looks for the following two cases:

Case 1. If there are LSH codes that are significantly different from the rest of the LSH codes: Based on the hamming distance between pairs of LSH codes and the similarity threshold $\Theta_{DL}$, the data aggregator determines that the compared pair of LSH codes are similar. If a LSH code is found to be similar with another LSH code, then its support count is increased by 1. The LSH codes that have a support count which is less than predetermined $minSup_{local}$ are labeled as local outliers. These local outliers, however, might be affected by the events that occurred in the neighboring clusters. Therefore, neighboring data aggregators exchange their local outlier lists among them to determine if these outliers can improve their support count. Each data aggregator compares LSH codes of its neighboring local outliers with its cluster’s LSH codes and updates their support counts. Neighboring data aggregators exchange the updated support counts of local outliers. Data aggregators check the updated support count of their local outliers and they label the local outliers that have a updated support count less than $minSup_{group}$ as outliers.

Case 2. If there are LSH codes that are exactly the same: During the comparison LSH code pairs, data aggregators also find out the sensor nodes that sent exactly the same LSH codes. In other words, data aggregators discover the sensor nodes that have the same data. This information is particularly useful to eliminate redundant data transmission from sensor nodes to the data aggregator. If there are more than one sensor nodes that have the same LSH code, then the data aggregator selects only one sensor node among them to send its actual data thereby reducing data transmission amount.

C. Phase 3. Data Aggregation

At the end of the Phase 2, the data aggregator has the list of outliers and the sensor nodes that have the same LSH codes. Based on this information, the data aggregator decides the sensor nodes that should send their actual data for data aggregation as follows. The data aggregator first eliminates the outliers, then it determines the sensor nodes that have distinct LSH codes and request only one sensor node to send the actual data for each distinct LSH code. Only requested sensor nodes send their data to the data aggregator and the data aggregator does not accept data from any other sensor nodes. This process ensures that (i) no outlier data is included in the aggregated data and (ii) there is no redundant data transmission from sensor nodes to the data aggregator. The data aggregator aggregates received data and send aggregated data to the base station.
data aggregation accuracy is affected by the FTDA’s outlier detection performance. If FTDA eliminates all the outliers in the network then the data aggregator does not receive any false data resulting in 100% correct data aggregation results. As seen from Figure 3, the percentage of false data in the networks increases, some of these false data can have sufficient minSupport values and are not labeled as outlier. As shown in previous subsection, wider Θ similarity threshold angels reduce the outlier detection performance of FTDA. Figure 3 also reflects this observation, the wider Θ results in reduced data aggregation accuracy due to undetected outliers.

VI. CONCLUSION

This paper presents a fault tolerant data aggregation scheme that eliminates the false data sent by malfunctioning and/or compromised sensor nodes. To prolong the lifetime of the network by saving energy, a Locality Sensitive Hashing (LSH) based in-network outlier detection technique is used. The simulation results show that the proposed scheme, FTDA, is able to detect outliers in most cases. As a result, FTDA reduces the number of false data transmissions thereby increasing the data aggregation accuracy.

REFERENCES