Research Article
Prediction-Based Filter Updating Policies for Top-\(k\) Monitoring Queries in Wireless Sensor Networks

Jiping Zheng, 1,2 Hui Zhang, 1 Baoli Song, 1 Haixiang Wang, 1 and Yongge Wang 1

1 College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, No. 29 Yudao Street, Baixia District, Nanjing 210016, China
2 State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

Correspondence should be addressed to Jiping Zheng; jzh@nuaa.edu.cn

Received 18 September 2013; Revised 18 January 2014; Accepted 29 January 2014; Published 13 April 2014

Academic Editor: Jianliang Xu

Copyright © 2014 Jiping Zheng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Processing top-\(k\) query in an energy-efficient manner is an important topic in wireless sensor networks. Redundant data transmitting between base station and sink node is avoided by installing filters on sensor nodes; thus, communication overhead between base station and sensor nodes is decreased. However, existing algorithms such as FILA, and DAFM consume much energy when updating the filter window. In this paper, we propose a new top-\(k\) algorithm named PreFU which is based on prediction models to update window parameters of filters. PreFU can predict the next \(s\) step sensor values based on time series predicting models which can be built by historical data. By estimating the cost of updating window parameters based on predicted sensor values, updates of filter window parameters can be reduced. Thus, the cost of updating window parameters is decreased. Experimental results show that our PreFU algorithm is more energy-efficient than existing algorithms while guaranteeing the accuracy of top-\(k\) query results.

1. Introduction

Despite all progress within research, energy consumption is still a major issue for wireless sensor networks [1, 2]. In addition, the quantity of generated data is large and dense. However, users are always interested in max or min \(k\) objects among them. Thus, top-\(k\) query processing is in high demand for many applications in uncertain databases and relational databases [3–5]. The top-\(k\) query in wireless sensor networks is different from general queries that data itself on sensor node could not determine whether it would be in final results. An intuitive way is that top-\(k\) query results will be determined by base station after collecting data from all sensor nodes. For our top-\(k\) queries, \(k\) is not restricted and the query determines the \(k\) highest observed values. The query also determines the full set of nodes that report the \(k\) highest values and is executed periodically starting at some point in time and reporting values for a number of subsequent rounds. The centralized query processing produces a large amount of communication cost and wastes lots of energy. So, how to process query in an energy-efficient manner is an important topic in wireless sensor networks, which can supply top-\(k\) results to users by minimizing the energy consumption. Generally speaking, query processing in wireless sensor networks is essentially different from that in traditional databases. A wireless sensor network containing \(N\) sensor nodes can be viewed as a distributed system, while this special distributed system is different from the general distributed system for there is not any single powerful node serving as the collection center to collect data from all the sensors. Each sensor transmits its data to the base station through multihop relays, which consumes energy for each data transmission. From another aspect, as the major optimization objective for query processing in wireless sensor networks, the network lifetime is determined not only by the total energy consumption of all sensors but also by the maximum energy consumption among the sensors. The sensors near to the base station consume more energy than the others, because they relay the data for the others and they will exhaust their batteries first. Once they run out of energy, the rest of the sensors will be disconnected from the base station, no matter how much residual energy left the rest of the sensors. Consequently,
the network is no longer functioning even if the total energy consumption per query is reasonably small. Hence, how to evaluate queries effectively and efficiently in wireless sensor networks poses great challenges.

A typical solution for answering top-$k$ queries in wireless sensor networks is by making the use of filters. Filters are broadcasted into the network and used by individual sensor node to decide whether its value is relevant to compute the query result. FILA (filter-based monitoring approach) [6] is an energy-efficient approach among these algorithms. The basic idea is to install a filter at each child sensor node, which avoids redundant data from transmitting to parent nodes or base station. However, the approach consumes much energy on updating filters. Mai et al. [7, 8] proposed DAFM algorithm according to FILA. The algorithm predicts the next sensed values of sensor nodes by linear regression model. Then, base station decides whether to update filtering windows based on predicted benefits. When data is in complicated distributions and varies widely over time, the predicted performance of linear regression is unsatisfactory such that the performance of DAFM algorithm gets worse. The contributions of this paper are summarized as follows.

(i) We adopt a powerful time series model ARIMA to predict next $s$ steps sensor values based on historical sensor data. The ARIMA time series model is suitable for sensor data due to the temporal correlations. Our proposed PreFU approach focuses on reevaluation when filters need updating. PreFU introduces prediction mechanism to achieve less energy consumption compared to eager filter update and lazy update policies in FILA approach. Also, compared to DAFM approach, our PreFU approach guarantees smaller mean squared error and thus can perform effectively (Algorithm 1).

(ii) Instead of one-step-forward prediction of DAFM approach, we adopt adaptive $s$-step prediction based on prediction errors. To select suitable $s$ values based on specified sensor data, PreFU outperforms existing approaches.

(iii) Extensive experiments are conducted to evaluate proposed PreFU approach by using real data traces. The results show that our PreFU approach outperforms DAFM, FILE-e, and FILA-l in terms of both energy consumption and network lifetime under single hop and multihop network configurations.

The rest of this paper is organized as follows. In Section 2, we discuss related work. Typical filtering methods for top-$k$ query processing are introduced in Section 3 and our proposed PrePU algorithm is provided in Section 4. In Section 5, extensive simulations are conducted to show the efficiency and accuracy of the proposed method. Finally, we conclude this paper in Section 6.

2. Related Work

A naive implementation of monitoring top-$k$ query is to use a centralized approach in which all sensor readings are periodically collected by the base station, which then computes the top-$k$ results set directly. However, a wireless sensor network can be viewed as a distributed network which consists of lots of energy-limited sensor nodes, and communication cost is the main energy consumption. Therefore, there is no doubt that the centralized approach will consume extra energy because of the transmission of massive data. In order to reduce the communication cost in data collection, Madden et al. proposed an in-network aggregation technique, known as TAG [9], which keeps unavailable data from transmission compared with centralized algorithms. However, this approach incurs unnecessary updates in the network and is not really energy-efficient.

Wu et al. [6] propose FILA approach for top-$k$ monitoring, which is to install a filter on each sensor node to filter out unnecessary data that are not contributed to final results. Reevaluation and filter setting are two critical aspects that ensure the correctness and effectiveness. When the new filters are different from the old ones maintained by sensor nodes themselves, base station needs to update them. But, when sensing values on nodes vary widely, base

Algorithm 1: PreFU algorithm.

```
(1) top-$k$ ← Query($k$) ⊳ Initialize top-$k$ set after collecting data from all sensor nodes
(2) if Get new values on base station then
(3) $N_u$ ← reevaluation() ⊳ Obtain sensor node set in which the filter window may be updated
(4) end if
(5) step ← 0
(6) repeat
(7) for $i = 1$ to $|N_u|$ do
(8) $v_{pre} ← predict(N_i)$ ⊳ Using ARIMA models to predict
(9) end for
(10) step ← step + 1
(11) Calculate $C_u, C_{update}, C_{old}$ for $s$-step-ahead prediction
(12) until step = $s$
(13) if $C_u + C_{update} < C_{old}$ then
(14) Update-Window() ⊳ Update windows in for nodes in $N_u$
(15)
```

(3) International Journal of Distributed Sensor Networks
(1) top-$k$ ← Query($k$) ⊳ Initialize top-$k$ set after collecting data from all sensor nodes
(2) if Get new values on base station then
(3) $N_u$ ← reevaluation() ⊳ Obtain sensor node set in which the filter window may be updated
(4) end if
(5) step ← 0
(6) repeat
(7) for $i = 1$ to $|N_u|$ do
(8) $v_{pre} ← predict(N_i)$ ⊳ Using ARIMA models to predict
(9) end for
(10) step ← step + 1
(11) Calculate $C_u, C_{update}, C_{old}$ for $s$-step-ahead prediction
(12) until step = $s$
(13) if $C_u + C_{update} < C_{old}$ then
(14) Update-Window() ⊳ Update windows in for nodes in $N_u$
(15) end if

(iii) Extensive experiments are conducted to evaluate proposed PreFU approach by using real data traces. The results show that our PreFU approach outperforms DAFM, FILE-e, and FILA-l in terms of both energy consumption and network lifetime under single hop and multihop network configurations.

The rest of this paper is organized as follows. In Section 2, we discuss related work. Typical filtering methods for top-$k$ query processing are introduced in Section 3 and our proposed PrePU algorithm is provided in Section 4. In Section 5, extensive simulations are conducted to show the efficiency and accuracy of the proposed method. Finally, we conclude this paper in Section 6.

2. Related Work

A naive implementation of monitoring top-$k$ query is to use a centralized approach in which all sensor readings are periodically collected by the base station, which then computes the top-$k$ results set directly. However, a wireless sensor network can be viewed as a distributed network which consists of lots of energy-limited sensor nodes, and communication cost is the main energy consumption. Therefore, there is no doubt that the centralized approach will consume extra energy because of the transmission of massive data. In order to reduce the communication cost in data collection, Madden et al. proposed an in-network aggregation technique, known as TAG [9], which keeps unavailable data from transmission compared with centralized algorithms. However, this approach incurs unnecessary updates in the network and is not really energy-efficient.

Wu et al. [6] propose FILA approach for top-$k$ monitoring, which is to install a filter on each sensor node to filter out unnecessary data that are not contributed to final results. Reevaluation and filter setting are two critical aspects that ensure the correctness and effectiveness. When the new filters are different from the old ones maintained by sensor nodes themselves, base station needs to update them. But, when sensing values on nodes vary widely, base
station needs updating filters to related nodes frequently which leads to large scales of updating cost and makes the performance of algorithm worse. Mai et al. [8] propose DAFM approach which aims to reduce the communication cost of sending probe messages in reevaluation aspect as well as the transmission cost in filter updating in FILA. DAFM approach predicts the next value of sensor nodes whose values are whether or not out of filtering windows by linear regression model. To some extent, this approach decreases the cost of filter updating. However, the sensed values are affected by various factors; the performance is worse when linear regression model was used to complicate data.

Besides these, filter-based and aggregation are two main strategies at present that they cope with each other to process top-\(k\) query in wireless sensor networks. Chen et al. propose QF (quantile filter, QF) [10] approach which treats sensing values and its sensor as a point. A top-\(k\) query is to return \(k\) points with highest sensing values. The goal of algorithm is to decrease energy consumption and prolong the lifetime of network which is not only minimizing the total energy consumption but also consuming less energy on each node. Liu et al. [11] propose a new cross pruning (XP) aggregation framework for top-\(k\) query in wireless sensor networks. There is a cluster-tree routing structure to aggregate more objects locally and a broadcast-then-filter approach in the framework. In addition, it provides an innetwork aggregation technique to filter out redundant values which enhance in-network filtering effectiveness. Abbasi et al. [12] proposed MOTE (model-based optimization technique) approach based on assigning filters on nodes by model-based optimization. Nevertheless, it is an NP-hard problem on how to get optimal filter settings for top-\(k\) set.

In addition, there are other related works about how to process top-\(k\) query in wireless sensor networks. Yeo et al. propose a novel technology called PRIM (priority-based top-\(k\) monitoring) which relied on the semantic of top-\(k\) query [13]. Its basic idea is to gather data according to priority; that is to say, the higher readings are collected earlier. Cho et al. propose POT algorithm (partial ordered tree) which considers the space correlation to maintain \(k\) sensor nodes with highest sensing values [14]. Michel et al. propose a framework KLEE to process top-\(k\) query [15] which allows for trade-off efficiency against result quality and bandwidth saving against communications. Different from the above, Silberstein et al. propose a sampling-based approach to evaluate approximate top-\(k\) queries in wireless sensor networks [16].

Energy-efficiency is a critical issue in wireless sensor networks and also an important indicator to evaluate the effectiveness and practicality of algorithm. In recent years, some researchers introduced time series models to wireless sensor networks. Tulone and Madden [17] apply autoregressive models to data collection in wireless sensor networks. The basic idea is to build a model on base station and each sensor node. When base station predicts the values of nodes until outlier readings produced, the nodes send sensing readings to base. When a reading is not properly predicted by the model, models are relearned to adapt changes. They are approximate readings collected by the approach. [18] is similar to [17]; both of them are approaches based on time series predicting in wireless sensor networks. The main difference is that the previous one is based on ARIMA. Although time series have been applied to wireless sensor networks, they just utilize to minimize energy consumption on data collection, not applied to top-\(k\) query processing. In this paper, we propose a novel top-\(k\) monitoring algorithm PreFU that combined the time series model ARIMA with FILA approach which avoids unnecessary filter updating cost and minimizes sensor node energy consumption.

3. Filtering Method for Top-\(k\) Query Processing

A typical system architecture of a wireless sensor network includes a base station and a number of sensor nodes. The base station has enough energy while sensor nodes are powered by batteries and energy-limited. When the base station is beyond a sensor node’s radio coverage, data are sent to base station via other sensor nodes through multiple jumps. Otherwise, sensed data are sent to base station directly.

Sensor nodes sense the physical phenomenon at a fixed sampling rate, such as temperature, humidity and light. When receiving the top-\(k\) query request, base station starts query and returns the results set to users at regular intervals. Assume that the sensed value on node \(N_i\) is \(v_i\), a top-\(k\) query is to return the ordered list of sensor nodes \(R\) with the highest readings at every epoch; that is, \(R = \langle N_1, N_2, \ldots, N_k \rangle, \forall i < j, v_i \geq v_j, \text{ and } \forall i \neq j (i = 1, 2, \ldots, k), v_i \leq v_k\). The results are maintained by base station and returned to users finally. The goal in this paper is to prolong the lifetime of wireless sensor networks by minimizing the overall energy consumption.

Initially, after collecting the readings from all sensor nodes, the base station sorts the readings and gets the initial top-\(k\) result set. The base station computes a filtering window \([u_i, u_j]\) for each node based on the initial top-\(k\) result set. Then, it sends these windows to corresponding sensor nodes. At the next sampling epoch, if the value on node \(N_i\) is within \([u_i, u_j]\), node \(N_i\) need not update its value maintained by base station. Otherwise, updating request is sent to base station. The base station will then reevaluate the top-\(k\) result and adjust the filter settings for influenced sensor nodes. According to different updating strategies, the base station sends the new filters to relevant sensor nodes based on updating strategies. In FILA, they provided two strategies: lazy filter update (FILA-L) and eager filter update (FILA-E).

In order to ensure the correctness of the algorithm, the space which is formed by all filtering windows should be continuous; that is, we set \(u_{i+1}\) to be equal to \(u_i\). FILA devises two filter setting approaches: uniform filter setting and skewed filter setting. In this paper, we adopt uniform filter setting; take Figure 1 as an example; the filtering windows of a top-\(k\) result are set by the following equation:

\[
u_{i+1} = l_i = \frac{v_i + v_{i+1}}{2} \quad (1 \leq i \leq k).
\]

To maximize the filtering capability, the upper bound of top-1 is set to +\(\infty\); the lower bound of nontop-\(k\) nodes are set to −\(\infty\). Then, the filtering window of nontop-\(k\) nodes
which are ranked at k-th is \([−∞, l_k]\). Seen from the above, the base station just needs maintaining \(k + 1\) filtering windows as shown in Figure 1.

To some extent, FILA avoids redundant data from transmission and saves energy. However, there is massive unnecessary energy consumption. Assume that, after sampling in epoch \(t_i\), the filtering windows are shown in Figure 2.

In Figure 2(a), at epoch \(t_i\), before sampling top-3 has the value of \([v_1, v_4, v_3]\), and the results set is \([v_1, v_4, v_3]\) after sampling as shown in Figure 2(b); that is, the value \(v_3\) jumps out of its filter and falls into the filtering window installed on \(N_4\). The base station needs to adjust the filtering windows for relevant nodes which are shown in Figure 2(c). However, after sampling in epoch \(t_{i+1}\) as shown in Figure 2(d), the value \(v_3\) changes again and jumps into the frontal window. In FILA, fluctuation of data will cause frequent updates of the filter windows which will incur large scale of communication cost. Mai et al. [8] proposed an updating algorithm based on prediction which determines whether to update the filters according to the possible cost on updating. The algorithm decreases the cost on updating the filters to some extent. However, this approach is limited by the prediction performance of linear regression model.

In addition, from Figure 2, the values on the nontop-\(k\) vary in a small range, their filtering windows are useful to filter out irrelevant sensor nodes. However, the upper bound of nontop-\(k\) nodes is determined by the values of \(k\)-th and \((k + 1)\)th sensor nodes; when the value which is ranked at \(k\)-th varies, the filters on nontop-\(k\) will be affected directly.

For a wireless sensor network that generally consists of large number of sensor nodes, \(k\) is usually small in a top-\(k\) query while a large number of sensor nodes fall into nontop-\(k\) node set. If the algorithm updates the filters once reevaluating the top-\(k\) results, it will consume more energy on updating the filters of nontop-\(k\) nodes.

Inherent defects of FILA and DAFM need better mechanisms to avoid too much communications while updating windows. So, we propose a new updating algorithm PreFU based on prediction by autoregressive integrated moving average (ARIMA) models. The algorithm evaluates the possible communication cost based on the \(s\)-step-ahead predicted values; usually \(s\) is smaller than 10. Then, the base station decides whether or not to update relevant filtering windows.

4. PreFU Approach

In FILA, when setting of the new filtering windows for nodes is changed after reevaluating the top-\(k\) results, the filters on nontop-\(k\) nodes are affected, and the frequent fluctuation of data will lead to unnecessary communication cost on updating windows. Considering two aspects above, by updating approach based on prediction, the algorithm decides whether to send the new filtering window parameters to corresponding nodes. In fact, we could not get the exact future values for each sensor node, but it can be obtained by prediction. In this paper, our PreFU approach predicts the next value(s) by ARIMA models.

4.1. ARIMA Model. ARIMA is a time series predicted model which predicts the next value(s) according to historical
data. A typical ARIMA model consists of three components: AR (auto regressive), I (the integrated), and MA (moving average). AR is a linear regression of the current value of the time series against one or more prior values of the series, which captures the relationship between current value and the latest \( p \) historical values. If there is a time series \( Z_t \), then the value at time \( t \) is represented by the latest \( t - 1 \) historical values, and the equation is shown as below:

\[
Z_t = \varphi_1 Z_{t-1} + \varphi_2 Z_{t-2} + \cdots + \varphi_p Z_{t-p} + \epsilon_t,
\]

Equation (2) is an autoregressive model and \( p \) is the order of the model. The parameters of the model are \( \varphi_1, \varphi_2, \ldots, \varphi_p \) and \( \epsilon_t \) is a white noise series. Since usually a time series may receive random shocks in a noisy environment, the MA is introduced to capture the influence of random shocks to the future. We call it moving average model MA(\( q \)), in which random errors at time \( t \) and \( t - 1 \) in the time series satisfy a linear regression model, as shown as below:

\[
Z_t = \theta_1 \varphi_{t-1} + \theta_2 \varphi_{t-2} + \cdots + \theta_q \varphi_{t-q} + \epsilon_t,
\]

where \( \theta_1, \theta_2, \ldots, \theta_q \) are moving average parameters of model which will be estimated. In general, the orders of AR or MA are high in order to describe the dynamic structure adequately. Without integrated components in ARIMA, the model is simplified as follows:

\[
Z_t = \varphi_0 + \sum_{i=1}^{p} \varphi_i Z_{t-i} + \epsilon_t - \sum_{j=1}^{q} \theta_j a_{t-j},
\]

where \( \varphi_0 \) is an initialized constant. However, ARMA model usually assumes that data is stationary; that is, the statistical properties of data do not change over time. This assumption does not hold in most real data series; therefore, the integrated term is introduced to remove the impact of nonstationary data by differencing. The time series is satisfied with ARMA model after several times of differencing. In general, first-order differencing is sufficient. Therefore, a time series is represented by an ARIMA (\( p,d,q \)) model that the time series is stationary after \( d \) times of differencing. And \( p \) represents the amount of historical data with successive timestamps; \( q \) represents the number of the latest random shocks on time series.

4.2. Time Series Model-Driven Prediction. Prediction by ARIMA model usually consists of two steps. The first step is model identification and parameter estimation. The second is prediction.

4.2.1. Model Identification and Parameter Estimation. Each parent node builds an ARIMA model to predict the next timestamps; \( \gamma_l \) represents the number of the latest random errors at time \( l \). Without integrated components in ARIMA, the model is simplified as follows:

\[
\gamma_l = \frac{\sum_{t=1}^{T} (Z_t - \bar{Z}) (Z_{t+l} - \bar{Z})}{\sum_{t=1}^{T} (Z_t - \bar{Z})^2}.
\]

\( T \) is the number of samples; \( l \) is the distance of interval; in general, \( l = 20 \). \( Z \) is used to estimate the expectation of time series which describes the average value in arithmetic. PACF describes the conditional correlation between \( Z_t \) and \( Z_{t-q} \), when given time series of \( Z_{t-1}, Z_{t-2}, \ldots, Z_{t-1+q} \). The degree of correlation is measured by \( \varphi_0 \) and is estimated by partial autocorrelation parameters. Consider

\[
\varphi_0 = \frac{\sum_{t=1}^{T} Z_t \sum_{i=1}^{T} Z_{t-i}}{\sum_{t=1}^{T} Z_t^2}.
\]

We can get the estimated values of \( p \) and \( q \) according to PACF and ACF. That is, the sample PACF cuts off at lag \( p \). As for \( q \), we can get it by ACF. For a time series \( Z_t \) with \( \gamma_l \), if \( \gamma_l \neq 0 \) when \( l \leq q \) and \( \gamma_l \) when \( l > q \), then the order of MA is \( q \). However, we just get the estimated \( p \) and \( q \); they can be decided by information criteria called AIC (Akaike information criterion). AIC criteria find the minimum \( p^* \) and \( q^* \) to minimize the value of AIC. We view \( p^* \) and \( q^* \) as the optimal estimated values of \( p \) and \( q \). The computation of AIC is shown as below:

\[
\text{AIC} = \ln \hat{\sigma}^2 + \frac{2(p + q)}{T}.
\]

\( T \) is the sample size; \( \hat{\sigma}^2 \) is the maximum-likelihood estimate of \( \sigma^2 \). Consider

\[
\hat{\sigma}^2 = \frac{\sum_{t=1}^{T} (Z_t - \sum_{j=1}^{c} \varphi_{ij} \times Z_{t-j})^2}{T - c}.
\]
After order determination of the ARMA model, we estimate parameters \(\{q_0, q_2, \ldots, q_p\}\) by the conditional least-squares method. Now, we can predict the next value of threshold by ARIMA model. In this procedure, the order of model will not be changed, but the parameters of the model are self-learning while processing top-\(k\) queries which ensures that prediction error is acceptable.

4.2.2. Predicting. As illustrated by (1) in Section 4.2, suppose that we are at time epoch \(h\); \(Z\) is a time series formed by previous epochs just before \(h\); one-step-ahead predicted value \(v_{\text{pre}}\) of the time series is calculated as below:

\[
v_{\text{pre}} = E(Z_{h+1} \mid Z_h, Z_{h-1}, \ldots, Z_1) = q_0 + \sum_{m=1}^{p} q_m Z_{h+1-m} - \sum_{m=1}^{q} \theta_m \alpha_{h+1-m}.
\]

4.3. Evaluation of the Cost. Predicting the next \(s\) values for each node in \(N_u\), the algorithm evaluates the possible cost adopting the new and the old filtering windows separately. We denote the cost by \(C_{\text{new}}\) and \(C_{\text{old}}\). The cost is referred to the number of communications caused by the fact that the values of nodes violate filters at the next sampling epoch. Only when \(C_{\text{new}} < C_{\text{old}}\), the base station sends the new filtering windows to nodes in \(N_u\).

Let \(v_{i, \text{pre}}\) be \(j\)th step prediction value of node \(i\) \((N_i \in N_u)\); we calculate \(C_{\text{old}}\) and \(C_{\text{new}}\) as follows:

\[
C_{\text{old}} = \sum_{i=1}^{N} \sum_{j=1}^{s} \delta_{i,j} \\
C_{\text{new}} = C_e + C_{\text{update}} = |N_u| + \sum_{i=1}^{N} \sum_{j=1}^{s} \delta'_{i,j},
\]

\[
\delta_{i,j} = \begin{cases} 1 & v_{i, \text{pre}} \notin [l_i, u_i] \\ 0 & \text{else} \end{cases} \\
\delta'_{i,j} = \begin{cases} 1 & v_{i, \text{pre}} \notin [l_i', u_i'] \\ 0 & \text{else} \end{cases}
\]

where \(C_e\) is the cost of updating filter windows for each sensor node in \(N_u\) and for \(s\)-step-ahead prediction, each sensor node in \(N_u\) just needs to be updated only once; that is, \(C_e = |N_u|\). And \(l_i'\) and \(u_i'\) are computed in the same way as (1); that is,

\[
u_{i, t+1}' = \frac{v_{i, t}' + v_{i, t+1}' }{2} \quad (1 \leq i \leq k).
\]

If the value of the sensor node \(N_i\) violates the filter, \(v_i'\) uses the new value sent to the base station for calculation. Otherwise, \(v_i' = v_i\), that is, using old value for calculation.

In our PreFU algorithm, if the base station updates the filters to nodes, the cost is composed of two aspects: the cost of updating the new filters and the cost of sending the updated values violating the filters at the next \(s\) step sampling to base station, denoted as \(C_{\text{update}}\) and \(C_e\), respectively. If the base station need not to update the filters, the cost of using the old filters at the next sampling epoch is referred to as \(C_{\text{old}}\). When \(C_{\text{new}} < C_{\text{old}}\), the base station updates the new filters. Otherwise, all nodes adopt the old filters.

5. Experiments

5.1. Experiment Settings. We evaluate the performance of the proposed algorithm PreFU by MATLAB simulations. The experimental data is from Intel Lab [19]. We adopt temperature, humidity and light data on March 1st, 2004, as experimental data. The data were collected from 54 sensor nodes, and data are collected every 31 seconds. In the paper, we treat the average of every two sample values as one.
We compare our proposed PreFU algorithm with FILA-E, FILA-L, and DAFM under uniform filter setting [6] for each window. We view the times of communication for a top-\(k\) query as the measurement with different \(k\). Considering that the energy consumption of sending is different from receiving, we simulate Mica2Dot sensor node which the energy consumption of sending is 0.37 times of receiving in this paper. The network lifetime is defined as time duration before the first sensor node runs out of power [6]. We adopt 2 AAA batteries with 15120 J for total energy capacity while the energy to establish a connection is 0.645 J and sending a byte data consumes 0.0144 J.

\[5.2.\text{Experiment Results}\]

\[5.2.1.\text{Comparison of Prediction Performance in Different Models.}\] We learn the ARIMA(2,0,2) utilizing historical data in the experiment. The parameters on different nodes are maintained by the base station dynamically. Figure 3 shows the predicted performance comparison of two models for \(s\)-step-ahead predicting. The prediction performance of ARIMA models is prior to linear regression model. Using ARIMA model to predict, the error rate is much smaller than linear regression model when \(k = 5, 6, 7,\) and 8. We set \(s\) to 5; that is, for every 5-step-ahead predicting we check whether to update the window parameters for each sensor.
5.2.2. Comparison of Four Algorithms in Different Network Structures and Data Distributions. First, the different network structures may affect the performance of algorithm. We evaluate the effectiveness of algorithm based on single-hop and multihop networks as shown in Figure 4 (the hop number is 2). Second, the different data distributions have affected the performance of algorithm. We evaluate the algorithm using following sensor attributes temperature, humidity, and light in different network structures to illustrate the efficiency of the proposed algorithm. Figures 5 and 6 show the results of our proposed algorithm and existing algorithms in different network structures and data distributions.

As shown in Figures 5 and 6, under two possible network structures single-hop and multihop and three different data temperature, humidity, and light, the proposed PreFU algorithm is prior to existing algorithms. Take Figure 5(b) as an example, when \( k = 10 \), PreFU algorithm is less 7000 communications than FILA-L and less 4000 communications than DAFM algorithm. The advantages lie in two aspects: first, \( k \) is smaller compared with the total number of nodes in the wireless sensor network. In addition, the nodes in the set of nontop-\( k \) share the same filter which is affected by the value on node ranked \( k \)th. Once the value on node ranked \( k \)th changed, the filtering windows in the set of nontop-\( k \) do not need to be updated. So, we update the filters-based \( s \)-step-ahead prediction which decrease the energy consumption. The second aspect is that the ARIMA model is prior to linear regression model on performance of prediction.
5.2.3. Comparison of Lifetime under Different Approaches.

We now consider the lifetime of DAFM, FILA-E, FILA-L, and our PreFU approaches under single-hop (Figure 7) and multihop (Figure 8) network distributions. In Figure 7, our PreFU approach is prior to other three approaches under temperature and humidity data. When \( k \) is small, our approach performs much better. However, under light data, when \( k \) is small, our PreFU approach is worse than DAFM approach. The reason is that light changes rapidly in real world and one-step-prediction will be better under small \( k \) values. When \( k \) increases, we should consider more sensor nodes as top-\( k \) results, our proposed PreFU approach performs better. In Figure 8, the same situation occurs and the reason is the same as above.

6. Conclusion

In order to cope with the problem that there is unnecessary updating cost in top-\( k \) query processing, we propose a new top-\( k \) query algorithm called PreFU which is based on time series prediction models. The algorithm evaluates the cost of updating the filters based on ARIMA prediction models which is built on the historical sensor data. Our PreFU
reduces the energy consumption and prolongs the life of network by avoiding unnecessary updating of filtering windows. The extensive experiments demonstrate the correctness and effectiveness of the proposed PreFU approach compared with FILA-E, FILA-E, and DAFM approaches.

Conflict of Interests
The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments
This work is supported by National Program on Key Basic Research Project of China (973 Program) under Grant no. 2014CB744900, the Fundamental Research Funds for the Central Universities of China under Grant no. NZ2013306, the Ph.D. Programs Foundation of Ministry of Education of China under Grant no. 20103218110017, Aeronautical Science Foundation of China under Grant no. 20115552030, and NUAA Research Funding of China under Grant NS2013089. The authors would also thank the reviewers for their helpful
comments and advices to improve the presentation of the paper.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, no. 4, pp. 393–422, 2002.

[2] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: a survey," Ad Hoc Networks, vol. 7, no. 3, pp. 537–568, 2009.

[3] I. F. Ilyas, G. Beskales, and M. A. Soliman, "A survey of top-\(k\) query processing techniques in relational database systems," ACM Computing Surveys, vol. 40, no. 4, article 11, 2008.

[4] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang, "Top-\(k\) query processing in uncertain databases," in Proceedings of the 23rd International Conference on Data Engineering (ICDE ’07), pp. 896–905, April 2007.

[5] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang, "Probabilistic top-\(k\) and ranking-aggregate queries," ACM Transactions on Database Systems, vol. 33, no. 3, article 13, 2008.

[6] M. Wu, J. Xu, X. Tang, and W.-C. Lee, "Top-\(k\) monitoring in wireless sensor networks," IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 7, pp. 962–976, 2007.

[7] H. T. Mai, Y. L. Ki, W. L. Yu, and H. K. Myoung, "Processing top-\(k\) monitoring queries in wireless sensor networks," in Proceedings of the 3rd International Conference on Sensor Technologies and Applications (SENSORCOMM ’09), pp. 545–552, June 2009.

[8] H. T. Mai, Y. W. Lee, K. Y. Lee, and M. H. Kim, "Distributed adaptive top-\(k\) monitoring in wireless sensor networks," Journal of Systems and Software, vol. 84, no. 2, pp. 314–327, 2011.

[9] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," in Proceedings of the 5th symposium on Operating systems design and implementation (OSDI ’02), pp. 131–146, 2002.

[10] B. Chen, W. Liang, R. Zhou, and J. X. Yu, "Energy-efficient top-\(k\) query processing in wireless sensor networks," in Proceedings of the 19th International Conference on Information and Knowledge Management (CIKM ’10), pp. 329–338, ACM, New York, NY, USA, October 2010.

[11] X. Liu, J. Xu, and W.-C. Lee, "A cross pruning framework for Top-\(k\) data collection in wireless sensor networks," in Proceedings of the 11th IEEE International Conference on Mobile Data Management (MDM ’10), pp. 157–166, May 2010.

[12] A. Abbasi, A. Khonsari, and N. Farri, "Mote: efficient monitoring of top-\(k\) set in sensor networks," in Proceedings of the 13th IEEE Symposium on Computers and Communications (ISCC ’08), pp. 957–962, Marrakech, Morocco, July 2008.

[13] M. H. Yeo, D. O. Seong, and J. S. Yoo, "Prim: priority-based top-\(k\) monitoring in wireless sensor networks," in Proceedings of the International Symposium on Computer Science and its Applications (CSA ’08), pp. 326–331, October 2008.

[14] Y. Cho, J. Son, and Y. D. Chung, "POT: an efficient top-\(k\) monitoring method for spatially correlated sensor readings," in Proceedings of the 5th International Workshop on Data Management for Sensor Networks (DMSN ’08), pp. 8–13, August 2008.

[15] S. Michel, P. Triantafillou, and G. Weikum, "KLEE: a framework for distributed top-\(k\) query algorithms," in Proceedings of the 31st International Conference on Very Large Data Bases (VLDB ’05), pp. 637–648, September 2005.
