Eleventh International Conference on Mobile Data Management

Online Time Interval Top-k Queries in Wireless Sensor Networks

Baichen Chen
Australian National University
Canberra, ACT, Australia
baichen@cs.anu.edu.au

Weifa Liang
Australian National University
Canberra, ACT, Australia
wliang@cs.anu.edu.au

Jeffrey Xu Yu
Chinese Univ. of Hong Kong
Shatin, N.T., Hong Kong
yu@se.cuhk.edu.hk

***Abstract*—Motivated by many applications, top-k query is a fundamental operation in modern database systems. Technological advances have enabled the deployment of large-scale sensor networks for environmental monitoring and surveillance purposes, efficient processing of top-k query in such networks poses great challenges due to the unique characteristics of sensors and a vast amount of data generated by sensor networks. In this paper, we first introduce the concept of time interval top-k query that is to return k highest sensed values from the sensory data generated within a specified time interval. We then propose a filter-based algorithm for time interval top-k query evaluation, which is capable to filter out nearly a half unlikely top-k data from transmission in comparison with a well known existing solution. We also develop a novel online algorithm for answering time interval top-k queries with various ks and time intervals one by one through maintaining a materialized view that consists of historical top-k query results. We finally conduct extensive experiments by simulations to evaluate the performance of the proposed algorithms on real sensory datasets. The experimental results show that the proposed algorithms outperform existing algorithms significantly to prolong the network lifetime.**

Keywords: *online time interval top-k queries, wireless sensor network, query optimization, energy conservation*

I. Introduction

In many application domains, top-k query is a fundamental query to search for the most important objects according to object ranking. Being different from those studies of top-k query in centralized databases, in this paper we focus on top-k query optimization in resource-constrained wireless sensor networks (WSNs). Technological advances have enabled the deployment of large-scale sensor networks consisting of thousands of inexpensive sensors in an ad-hoc fashion for a variety of environmental monitoring and surveillance purposes. During this course, a large volume of sensed data are needed to be aggregated within the network to respond to user queries. The wireless sensor network thus is treated as a virtual database by the database community [1]. However, query processing in WSNs is essentially different from it in traditional databases due to the unique characteristics imposed on sensors, e.g., slow processing capability, limited storage, and energy-limited batteries, etc. [7], which can be seen from several aspects. Firstly, to prolong network lifetime, the energy consumption is an optimization objective in WSNs, because the battery-powered sensors will quickly become inoperative due to the large quantity of energy consumption, and the network lifetime is closely tied to the energy consumption rate of the sensors. Secondly, a wireless sensor network that senses the data periodically can be viewed as a distributed stream system [2]. However, this special distributed stream system is different from the general distributed stream system because it is more expensive to obtain the sensed data from the sensors far away from the base station than those nearby. Finally, for query processing in WSNs, minimizing not only the total energy consumption but also the maximum energy consumption among the sensors is the optimization objective. Hence, how to evaluate queries effectively and efficiently in WSNs poses great challenges.

Previous studies of top-k query in distributed systems mainly focused on *the distributed top-k query*, which is to find the k objects with highest scores, assuming that each object is distributed in multiple nodes. In each node, the object has a local score. The score of the object is combined from all of local scores of the object by a given function [2], [6]. In this paper, we deal with a different top-k query which is identical to the one proposed in [13], [3]. That is, each sensor senses one or multiple numerical values from its vicinity, and each value and its generator sensor as well as the generation time is referred to as a *point* in the rest of this paper. A point p is represented by a tuple $(p.sid,\ p.tid,\ p.val)$, where $p.sid$ is the ID of generator sensor of point p, $p.tid$ is the time unit when point p is generated, and $p.val$ is the sensed numerical value. *The top-k query* in WSNs is to return the k points with the highest sensed values. On the other hand, a WSN should be able to support top-k query processing with different values of k and time intervals. We refer to such a top-k query as a *time interval top-k query*, denote by top-$(k, [t_s, t_e])$ which aims to identify k points with highest sensing values from a set of points generated within time interval $[t_s, t_e]$, where $t_s \leq t_e$.

Top-k query has been extensively studied in traditionally relational databases [4]. In recent years, the studies of top-k query in other databases like distributed databases [2] and data stream [5] have been exploited. However, due to essential difference of top-k query processing between these databases and wireless sensor networks, the proposed algorithms for these models cannot be applicable to WSNs. Previous studies investigated in-network aggregation in wireless sensor networks for energy optimization, e.g., the algorithms in [1], [11] are exploited for simple aggregation, and other algorithms are proposed to deal with the more complicated queries including skyline query [12] and distributed top-k query [6]. Several algorithms are proposed to solve the top-k query in WSNs that returns the k points with the highest sensed values [13], [3]. Silberstein *et al* [13] considered the top-k

978-0-7695-4048-1/10 $26.00 © 2010 IEEE
DOI 10.1109/MDM.2010.30

query evaluation problem by providing approximate solutions with high probability, based on a prediction model that is built on the samples of previous top-k query results. Wu *et al* [3] exploited the semantics of top-k query and proposed a filter-based maintenance algorithm (FILA) to maintain the current top-k points by assigning a dedicated filter for each sensor. The proposed algorithms by both Silberstein *et al* [13] and Wu *et al* [3] are centralized algorithms, which may not be suitable for the real distributed WSNs.

In this paper, we focus on the time interval top-k query optimization to maximize the network lifetime through striking the right balance between the total energy consumption and the maximum energy consumption. The main contributions of this paper are as follows. We first introduce new concepts of time interval top-k query and online time interval top-k queries with various ks and time intervals under sliding window environments in WSNs, where the points are dynamically generated and expired. We then propose an energy-efficient optimization framework for time interval top-k queries. We devise a filter-based localized evaluation algorithm for answering the top-k query on a snapshot dataset. We also deal with online time interval top-k queries by proposing a novel incremental algorithm for answering various time interval top-k queries in a streaming dataset on-the-fly. We finally conduct extensive experiments by simulations on real sensory datasets to evaluate the performance of the proposed algorithms. The experimental results show that the proposed optimal framework is very efficient and effective, in terms of various metrics.

The remainder of the paper is organized as follows. Section 2 introduces the cost model of WSNs and problem definitions, followed by introducing an existing top-k query evaluation algorithm for the benchmark purpose. Section 3 proposes an energy-efficient algorithm for time interval top-k query evaluation, and Section 4 devises a novel algorithm for online time interval top-k queries. Extensive experiments are conducted in Section 5 to evaluate the performance of the proposed algorithms, and the conclusions are given in Section 6.

II. Preliminaries

A. System Model

We consider a sensor network consisting of n stationary sensors randomly deployed in a region of interest, and a base station r with unlimited energy supply located at the center of the region. For the sake of simplicity, we assume that the topology of the sensor network is a spanning tree $\mathcal{T}$ rooted at the base station; otherwise, such a tree can be obtained by applying any spanning tree algorithm in the network like the one by TAG [1]. All sensors have identical transmission ranges, and they can communicate with the base station via one or multi-hop relays. We assume that the time dimension is infinite and the basic time unit is time step. At each time step, some rather than all sensors generate one or multiple new points. Assume that for a point p, each of $p.sid$, $p.tid$ and $p.val$ is represented by 4 bytes. Thus, point p is represented by 12 bytes in total. Each point has a fixed lifespan w, i.e., point p will expire at the time step $p.tid + w$. To transmit a message containing l bytes of data from a sensor to one of its neighbors, the amounts of transmission energy consumption at the sender are $\rho_t + R * l$ and the amounts of reception energy consumption at the receiver are $\rho_r + r_e * l$, where ρ_t and ρ_r are the sum of energy overhead on handshaking and transmitting and receiving the message header, R and r_e are the amounts of transmission and reception energy per byte, respectively. We assume that the computation energy consumption on sensors can be ignored, because it is several orders of magnitude less than that of the communication energy consumption [7].

B. Problem Definition

Given a point p, we say point p is *valid* at time step t if $t - p.tid \leq w$; otherwise, p is invalid. Denote by $P_t(v_i)$ the set of valid points at a sensor v_i and $P_t = \bigcup_{i=1}^{n} P_t(v_i)$ the set of valid points in the sensor network at time step t. Let $P_t(v_i)[t_1, t_2]$ and $P_t[t_1, t_2]$ be the subsets of $P_t(v_i)$ and P_t in which the points are generated between time steps t_1 and t_2. Denote by $Update_t(v_i)$ and $Expired_t(v_i)$ the sets of points at sensor v_i generated and expired at time step t, respectively. At each time step t, each sensor updates the set of points, i.e., $P_t(v_i) = P_{t-1}(v_i) \cup Update_t(v_i) - Expired_t(v_i)$. Therefore, $P_t = P_{t-1} \cup \bigcup_{i=1}^{n} Update_t(v_i) - \bigcup_{i=1}^{n} Expired_t(v_i)$. Because each point can only survive w time steps and all points generated before time step $t - w + 1$ must expire at time step t, $P_t = \bigcup_{i=1}^{n} \bigcup_{x=t-w+1}^{t} Update_x(v_i)$. The sensor network is thus regarded as a distributed stream system with a boundless streaming data. It is usually impossible to store all of the generated points at sensors due to their limited storage. Consequently, we here consider the time interval top-k query within sliding windows. A sliding window consists of w consecutive time steps and slides along the time dimension. At time step t, the sliding window contains all the points generated from $t - w + 1$ to t, and therefore the set of those points is P_t. *A time interval top-k query under a sliding window*, represented by top-$(k_t, [t_s, t_e])$, thus is to inquire the k_t points with the highest sensed values in set $P_t[t_s, t_e]$, where $t - w + 1 \leq t_s \leq t_e \leq t$. *Online time interval top-k query* is to return the result of each time interval top-k query with different values of k and time intervals issued at different time steps one by one.

Before we proceed, we briefly review an algorithm `Naive-`k for top-k query evaluation, in which each leaf sensor forwards its points to its parent. If a sensor contains k' ($\leq k$) points , then it forwards all the points to its parent; otherwise, it forwards its top-k points to its parent. In the end, the root identifies the top-k points from the collected points, which is the result of the top-k query.

III. Time Interval Top-k Query Evaluation Algorithm

In this section we propose a novel, filter-based localized algorithm for time interval top-k query evaluation on snapshot datasets. The basic idea of the proposed algorithm is that every sensor sorts its points in decreasing order of sensed values, and sends one of the values to its parent if the parent exists. The parent finds a filter by choosing one of the received values and broadcasts the filter to all its children. Each child sends those points whose values are no less than the filter to the parent. In the following we first provide the details of finding the filter,

followed by addressing whether a filter requires to be installed at a sensor. We finally present the proposed algorithm for time interval top-k query evaluation.

A. Filter Generation

Suppose that a time interval top-k query top-$(k_t, [t_s, t_e])$ is issued at time step t. For convenience, the points referred to in this section are the valid points generated from time step t_s to time step t_e unless otherwise specified. Consider a sensor v in the routing tree $\mathcal{T}$ with d_v children $u_1, \ldots, u_{d_v}$. $L(i) = \{p(i)_1.val, \ldots, p(i)_{l(i)}.val\}$ is the value set of points at u_i with $p(i)_{j_1}.val \geq p(i)_{j_2}.val$ if $1 \leq j_1 \leq j_2 \leq l(i)$, where $l(i)$ is the number of the points at u_i if $l(i) < k$. Otherwise, $l(i) = k$. In other words, there are at most k values in $L(i)$. $S(v) = \bigcup_{i=1}^{d_v}\{p \mid p.val \in L(i)\}$ is the set of potential top-k points at v from its children, and $|S(v)| \leq k * d_v$. In the following, the detail of finding the filter is described.

Each child u_i sends the median value of $L(i)$, $p(i)_{\lceil l(i)/2 \rceil}.val$ and $l(i)$ to its parent v, $1 \leq i \leq d_v$. The d_v received median values by sensor v are sorted in decreasing order and let $m_{i_1}, m_{i_2}, \ldots, m_{i_{d_v}}$ be the sorted sequence, where m_{i_j} is the median value from child u_{i_j}. The yth largest value m_{i_y} is chosen as the filter by sensor v, where $y = min\{e \mid \sum_{j=1}^{e} \lceil l(i_j)/2 \rceil \geq k\}$, which guarantees that there are at least k points in $S(v)$ whose values are no less than the filter m_{i_y}. Obviously the points in $S'(v) = \{p(i_a)_b \mid a \geq y, b \geq l(i_a)/2\}$ are smaller than m_{i_y} because $p(i_a)_b < m_{i_a} < m_{i_y}$ when $a \geq y$ and $b \geq l(i_a)/2$.

Having found the filter, sensor v broadcasts the filter to each child u_i, and each child u_i then sends the points whose values are no less than the filter to parent v, $1 \leq i \leq d_v$. We now analyze the filtering capability of the proposed filter. Compared to algorithm `Naive-`k in which all $|S(v)|$ points are transmitted to sensor v, at least $|S'(v)|$ points can be filtered out from transmission. The shedding ratio of data derived from the proposed filter is analyzed as follows.

The size of $S'(v)$ is $|S'(v)| = \sum_{j=y+1}^{d_v} \lceil l(i_j)/2 \rceil + \lceil l(i_y)/2 \rceil - 1 \geq \frac{1}{2}(|S(v)| - \sum_{j=1}^{y-1} l(i_j)) - 1$, then

$$\begin{aligned} \frac{|S'(v)|}{|S(v)|} &\geq \frac{\frac{1}{2}(|S(v)| - \sum_{j=1}^{y} l(i_j)) - 1}{|S(v)|} \\ &\geq \frac{\frac{1}{2}(|S(v)| - 2(k-1)) - 1}{|S(v)|} = \frac{1}{2} - \frac{k}{|S(v)|}. \end{aligned} \quad (1)$$

B. The Optimal Filter

As use of the filter at the children of sensor v can prune some unlikely top-k points in $S(v)$ from transmission, we now aim to prune as many points from $S(v)$ as possible, using an optimal filter. Consequently, we can significantly reduce the transmission and reception energy consumptions of sensor v through the reduction of data transmission. The rest is to find such an optimal filter.

Recall that sensor v is the parent of sensors $u_1, u_2, \ldots, u_{d_v}$. We further assume that each child knows how many siblings d_v it has; otherwise, this can be done through a broadcast from sensor v. To find the optimal filter, instead of transmitting the median value of $L(i)$, each sensor u_i transmits $p(i)_x.val$ that is the xth largest values in $L(i)$ to parent v, where x will be determined later, $1 \leq x \leq k$. Sensor v then sorts the received d_v values in decreasing order. Let $m_{i_1}, m_{i_2}, \ldots, m_{i_{d_v}}$ be the sorted sequence and $m_{i_j} \geq m_{i_{j+1}}$. Imagine that the points in $S(v)$ are arranged into a matrix $\{c_{a,b}\}_{k \times d_v}$ column by column, according to the ranks of their xth largest values of the points in the sorted sequence, i.e., $c_{a,b} = p(i_b)_a$ and $c_{x,b}.val = m_{i_b}$, $1 \leq a \leq k$, $1 \leq b \leq d_v$. Specifically, matrix $\{c_{a,b}\}_{k \times d_v}$ is constructed as follows (as shown in Fig. (1).

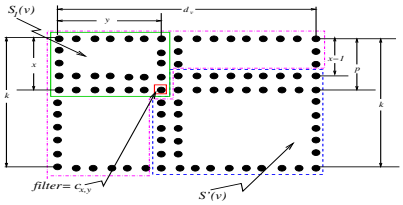


Fig. 1. The elements in $S(v)$ are partitioned by the element $c_{x,y}$

A point $c_{x,y}$ is identified such that $|S_1(v)| = x * y \geq k$, where $S_1(v)$ is the set of points in the top left corner of the matrix. Meanwhile, $|S'(v)|$ is required to be as large as possible where $S'(v)$ consists of the points in the bottom right corner of the matrix, because all the points in $S'(v)$ will be pruned from transmission. Thus, we aim to find such x and y that $|S'(v)|$ is maximized, where

$$\begin{aligned} |S'(v)| &= kd_v + k + d_v + xy - 1 - (d_v x + ky + x + y) \\ &\geq kd_v + d_v + 2k - 1 - (d_v x + ky + x + y), \end{aligned} \quad (2)$$

subject to $xy \geq k, 1 \leq x \leq k, 1 \leq y \leq d_v$. In other words, we aim to minimize the value $f(x,y) = (d_v x + ky + x + y)$ under the constraints. Since $|S_4(v)| \geq kd_v + k + d_v + k - 1 - f(x,y)$ and parameters k and d_v are given, $f(x,y)$ is minimized, when $x = \sqrt{\frac{k(k+1)}{d_v+1}}$ and $y = \sqrt{\frac{k(d_v+1)}{k+1}}$. In fact, both x and y are required to be positive integers. To do so, let $x_1 = \lceil \sqrt{\frac{k(k+1)}{d_v+1}} \rceil$ and $y_1 = \lceil \sqrt{\frac{k(d_v+1)}{k+1}} \rceil$, let $F(x,y) = kd_v + k + d_v + xy - 1 - (d_v x + ky + x + y)$ be the number of points in $S'(v)$ when $c_{x,y}.val$ is chosen as the filter. Then, sensor v computes x and y $(x * y \geq k)$ such that

$$\begin{aligned} F(x,y) = \max\{&F(x_1, y_1), F(x_1 - 1, y_1), \\ &F(x_1, y_1 - 1), F(x_1 - 1, y_1 - 1)\}, \end{aligned} \quad (3)$$

Fig. 1 shows the partition of points in $S(v)$ by the point $c_{x,y}$. The ratio of shedding data to the total data in $S(v)$ thus is

$$\begin{aligned} \frac{|S'(v)|}{|S(v)|} &= \frac{(k - x + 1)(d_v - y + 1) - 2}{kd_v} \\ &\geq \frac{(k - \lceil \sqrt{\frac{k(k+1)}{d_v+1}} \rceil + 1)(d_v - \lceil \sqrt{\frac{k(d_v+1)}{k+1}} \rceil + 1) - 2}{kd_v} \\ &\geq 1 + \frac{2}{d_v} + \frac{1}{k} - \frac{2\sqrt{(d_v+1)k(k+1)}}{kd_v} - \frac{1}{kd_v}, \end{aligned} \quad (4)$$

when $d_v \geq 2$.

C. The Filter-based Algorithm

Although installing a filter at a sensor may reduce unlikely top-k points from transmission, it also incurs extra energy

overhead on the filter finding and broadcasting. We now analyze this extra energy overhead as well as the energy saving brought by the filter. We only install the filter at those sensors that the energy saving exceeds the overhead on the filter installation.

For a given sensor v, if no filter is installed at each child u_i of v, u_i sends its top-$l(i)$ points to sensor v as `Naive-`k does, $1 \leq i \leq d_v$. As a result, the number of points transmitted to v is $|S(v)|$ and the total energy consumption by transmitting and receiving all $|S(v)|$ points is

$$E_{naive}(S(v)) = \rho_t d_v + 12|S(v)|R + \rho_r d_v + 12|S(v)|r_e. \quad (5)$$

If the filters are installed at the children of v, there are at least $|S(v)| * (\frac{1}{2} - \frac{k}{|S(v)|})$ points filtered out from transmission, thus, the total energy consumption is

$$E_{filter}(S(v)) = d_v(\rho_t + \rho_r + 8(R + r_e)) + \rho_t + 4R$$
$$+d_v\rho_r + d_v 4r_e + d_v(\rho_t + \rho_r) + (\tfrac{|S(v)|+2k}{2})(R + r_e), \quad (6)$$

where $d_v(\rho_t + 8R + \rho_r + 8r_e)$ is the total energy consumption of the children sending the values and $l(i)$s to sensor v and sensor v receiving the values from its children, $\rho_t + 4R + d_v(\rho_r + 4r_e)$ is the energy consumption of broadcasting the filter, and $d_v(\rho_t + \rho_r) + (\frac{|S(v)|+2k}{2})(R + r_e)$ is the sum of energy consumption that each child sends the points passing through the filter to sensor v.

what follows is to investigate the extra energy overhead on filter finding and the energy saving of a child of v. If there is no filter installed at a child u_i of v, u_i transmits its top-$l(i)$ points to sensor v and the energy consumption of u_i is

$$E_{naive}(L(i)) = \rho_t + l(i) * 12R; \quad (7)$$

otherwise, the energy consumption of u_i is

$$E_{filter}(L(i)) = 2\rho_t + 8R + \rho_r + 4r_e + l(i)/2 * 12R; \quad (8)$$

In Eq. (8), we assume that the expected number of points filtered out at sensor u_i by the filter is $l(i)/2$. Notice that in Eqs. (5) and (6) $|S(v)| = \sum_{i=1}^{d_v} l(i)$. In practice the value of $l(i)$ is not known by v beforehand. To this end, an approximate value $|S(v)_{app}|$ of $|S(v)|$ is proposed. Suppose that $desc(v)$ is the set of descendants of v and the number of descendants $|desc(v)|$ of v is given, then $|S(v)_{app}| = \sum_{i=1}^{d_v} \min\{k, |desc(u_i)|\}$. In other words, if a child contains more than k descendants, it transmits its top-k points to v; otherwise, the child transmits all points from its descendants to v. Obviously $|S(v)_{app}|$ can be obtained when the routing tree is built.

To guarantee that the installation of the filter at the children of sensor v will be beneficial, we thus have $E_{filter}(S(v)) < E_{naive}(S(v))$ and $E_{filter}(L(i)) < E_{naive}(L(i))$. Meanwhile, the shedding ratio of data will be no less than 0 only when $|S(v)| \geq 2k$ from Eq. (1). Combined with Eqs. (5), (6), (7) and (8) and replacing $|S(v)|$ by $|S(v)_{app}|$, we have

$$|S(v)_{app}| > \max\{\frac{\rho_t(d_v + 1) + 2\rho_r * d_v + (8d_v + 4)R}{12(R + r_e)}$$
$$+\frac{12d_v * r_e}{12(R + r_e)} + k,\ 2k\}; \quad (9)$$

and

$$l(i) > \frac{\rho_t + \rho_r + 8R + 4r_e}{6R}. \quad (10)$$

Having addressed the conditions of filter installation, we are ready to describe the proposed algorithm.

Suppose that an internal sensor v in the routing tree has d_v children $u_1,\ldots,u_{d_v}$, and each child u_i knows d_v and $|S(v)_{app}|$, which can be obtained by a broadcasting from sensor v when the routing tree is built. For a child u_i, if either condition (9) or (10) is not satisfied, u_i sends all of its top-$l(i)$ points to sensor v, and no filter is installed at u_i; otherwise, the proposed filter will be installed at sensor u_i as follows. Each child u_i sends the value of the xth point to the parent sensor when $|S(v)_{app}| = k * d_v$, and the parent broadcasts the yth largest value as the filter to each child, where x and y are the values to maximize $F(x, y)$ in (3). If $|S(v)_{app}| < k * d_v$, each child u_i sends its median value and $l(i)$ to the parent sensor v. Parent v then broadcasts the yth largest one among the d_v received values as the filter to its children, where $y = min\{e \mid \sum_{j=1}^{e} \lceil l(i_j)/2 \rceil \geq k\}$. Only the child sensor u_i with $l(i) > \frac{\rho_t+\rho_r+8R+4r_e}{6R}$ needs to receive the filter and then sends its points whose values are no less than the filter to parent v. The other children send all of their top-$l(i)$ points to v already.

IV. Online time interval Top-k Query Processing

So far the proposed algorithm has only been applied for top-k query evaluation on a snapshot dataset. To answer online queries, a naive approach is to evaluate the queries one by one, by employing the proposed evaluation algorithm. However, this approach will consume excessive energy because it is likely that most top-k points in previous top-k queries are still in the result of the current top-k query due to slow updates of points in the network. In this paper we propose an algorithm to evaluate the query incrementally by building a materialized view based on the historical query results. Intuitively, given a query top-$(k_t, [t_s, t_e])$, the base station sorts the points in the materialized view, which are generated within the time interval $[t_s, t_e]$ in decreasing order of their values. If there are at least k_t points in the view generated within the interval, the base station broadcasts the value of the k_tth point as the filter along with the query to the sensor network, and each sensor filters out the points with smaller values than the received filter. The proposed evaluation algorithm in the previous section is then applied to return the query results. Beyond this conceptually simple idea, several non-trivial issues remain to be answered. That is, how to maintain the materialized view such that the view contains k_t points generated in the unpredictable time interval of next query with high probability, how to decide the size of the materialized view, because it is prohibitive to store all the historical points in the view. In the following, we show how to maintain the materialized view based on a probability model, and answer online time interval top-k query using the materialized view.

A. The Probability Model

The materialized view at the base station consists of the points of previous top-k queries. The key to maintain the view is to calculate the expected number of points generated within

the time interval of the next query, referred to as $E(t_s, t_e)$. Having the result of the current query, the view will be updated by inserting the new points of the results into and removing the old ones from the view. The computation is based on a probability model because the exact time interval of the next query is not known beforehand. In the following, we describe the computation of $E(t_s, t_e)$.

Let t be the current time step. Denote by $V(t) = \bigcup_{t_i=t-w+1}^{t} v(t_i)$ the materialized view at time step t, where $v(t_i) = \{p \mid p.tid = t_i\}$ is the set of points generated at time step t_i, $t - w + 1 \leq t_i \leq t$. Let $Pr(c_1|c_2)$ be the conditional probability of event c_1. For a point p with $p.tid = t_i$, the probability of a point p generated in time interval $[t_s, t_e]$ is

$$\begin{aligned} Pr(p \in [t_s, t_e]) &= Pr(t_s < t_i, t_e > t_i \mid t_e > t_s) \\ &= \frac{t_i}{w} * \frac{w - t_i + 1}{w - t_s + 1} \geq \frac{t_i}{w} * \frac{w - t_i + 1}{w}. \end{aligned} \quad (11)$$

Therefore, the expected number of points in the view generated within the interval $[t_s, t_e]$ is

$$\begin{aligned} E(t_s, t_e) &= \sum_{t_i=t-w+2}^{t} (|v(t_i)| * Pr(p \in [t_s, t_e]) \\ &\geq \sum_{t_i=t-w+2}^{t} (|v(t_i)| * \frac{t_i}{w} * \frac{w - t_i + 1}{w}). \end{aligned} \quad (12)$$

We refer to $E_{min} = \sum_{t_i=t-w+2}^{t} (|v(t_i)| * \frac{t_i}{w} * \frac{w-t_i+1}{w})$ as the minimum expected number of the points generated within the time interval of the next query. We only consider the point generated from time step $t-w+2$, since the points generated at time step $t - w + 1$ will be expired at the next time step.

B. Online Time Interval Top-k Query Algorithm

We describe how to maintain the materialized review and online time interval top-k query algorithm as follows.

Assume that a query top-$(k_t, [t_s, t_e])$ is issued at time step t. The materialized view at the base station is $V(t)$. Assume that the expired points are removed from $V(t)$. Denote by $Vp(t_s, t_e) = \{p \mid t_s \leq p.tid \leq t_e, p \in V(t)\}$ the set of points in the view generated within the interval $[t_s, t_e]$ and the points in $Vp(t_s, t_e)$ are sorted in decreasing order of their values. If $|Vp(t_s, t_e| \geq k_t$, the value of the k_tth point in $Vp(t_s, t_e)$ is broadcast along with the query as the filter to the sensor network. The proposed algorithm for top-k evaluation is then applied on the set of remaining points with values being not small than the filter in $P(t)[t_s, t_e]$, to return the result of query top-$(k_t, [t_s, t_e])$, referred to as $Top(t)$. The set of points in $Top(t)$ that are not included by $V(t)$ is referred to as $New(t) = \bigcup_{t_i=t-w+2}^{t} new(t_i)$, where $new(t_i) = \{p \mid p.tid = t_i, p \in Top(t), p \notin V(t)\}$. We also exclude the points generated at time step $t-w+1$ because they will be expired at the next time step and will not contribute to the next query. Let k_{max} be the maximum value of k in all issued queries, which can be given beforehand. The base station calculates $E_{min} = \sum_{t_i=t-w+2}^{t} (|v(t_i)| * \frac{t_i}{w} * \frac{w-t_i+1}{w})$. If $E_{min} \geq k_{max}$, we will execute *replacement* operations until there is no point in $New(t)$ that can increase the value of E_{min}; otherwise, we will execute *insertion* operations and update the value of E_{min} of the new view until the updated E_{min} is not smaller than k_{max}, followed by executing *replacement* operations. The two operations are described as follows.

The *replacement* operation is to replace a point $p \in V(t)$ with a point $q \in New(t)$ if existent. Point p with $p.tid = t_1$ is chosen from $V(t)$ such that

$$p.val = min\{x.val \mid x \in v(t_1)\}, \quad (13)$$

$$\frac{t_1}{w} * \frac{w-t_1+1}{w} = min\{\frac{t_i}{w} * \frac{w-t_i+1}{w} \mid |v(t_i)| > 0\}. \quad (14)$$

Having chosen a point p, another point q with $q.tid = t_2$ is then chosen from $New(t)$ such that

$$q.val = max\{x.val \mid x \in new(t_2)\}, \quad (15)$$

$$\frac{t_2}{w} * \frac{w-t_2+1}{w} = max\{\frac{t_i}{w} * \frac{w-t_i+1}{w} \mid |new(t_i)| > 0\}, \quad (16)$$

$$\frac{t_2}{w} * \frac{w-t_2+1}{w} \geq \frac{t_1}{w} * \frac{w-t_1+1}{w}. \quad (17)$$

Note that t_1, t_2 and t_i in Eqs. (13)-(16) and Ineq. (17) are within $[t - w + 2, t]$. Having replaced point p by point q, the value of E_{min} will increase due to $\frac{t_2}{w} * \frac{w-t_2+1}{w} \geq \frac{t_1}{w} * \frac{w-t_1+1}{w}$. The replacement operations will terminate if there are no such a pair of points p and q meeting conditions (13)-(17).

The *insertion* operation is to insert a point $q \in New(t)$ being satisfied Eqs. (15) and (16) into $V(t)$ when $E_{min} < k_{max}$. After the insertion, E_{min} will be recalculated for the new materialized view. If the updated E_{min} is not smaller than k_{max}, the mentioned *replacement* operations will be executed; otherwise, another point in $New(t)$ meeting Eqs. (15) and (16) will be inserted into the view.

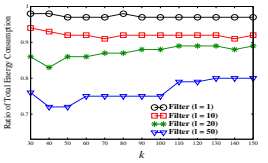
Having updated the materialized view based on the result of the current query, the filter for next query top-$(k_{t'}, [t_{s'}, t_{e'}])$ will be drawn from the updated view if there are at least $k_{t'}$ points generated within the time interval $[t_{s'}, t_{e'}]$.

V. Performance Study

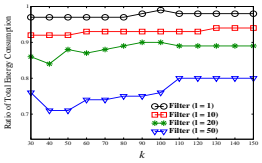
In this section, we evaluate the performance of the proposed algorithms in terms of the total energy consumption and the maximum energy consumption among sensors. The proposed algorithm for top-k query evaluation is referred to as `Filter`, while the one for online time interval top-k query processing is referred to as `View-Filter`. The performance of algorithm `Naive-`k is used for the benchmark purpose.

A. Experiment Setting

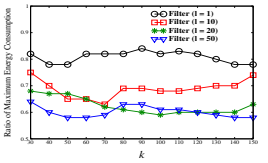
We assume that the sensor network is used to monitor a $100m \times 100m$ square region of interest. Within the region, n sensors are randomly deployed by the `NS-2` simulator [10] and the base station is located at the square center. We assume that all sensors have 5-meter transmission ranges and the topology of the sensor network is the TAG routing tree [1]. In our experiments, two sensor network instances consisting of 1,000 and 1,500 sensors are considered. In our experiments we adopt the transmission and reception energy consumption parameters of a real sensor MICA2 mote, where the energy consumption on transmitting and receiving a header and handshaking are $\rho_t = 0.4608\ mJ$ and $\rho_r = 0.1152\ mJ$, and the energy consumption of transmitting and receiving one byte are $R = 0.0144\ mJ$ and $r_e = 0.00576\ mJ$, respectively [8]. Each point p is represented by 12 bytes. The sensed points are drawn from a real dataset of temperature traces, collected by the Intel Berkeley Research Lab [9].



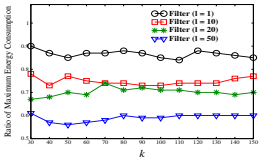
(a) The ratio of total energy consumption when $n = 1,000$



(b) The ratio of total energy consumption when $n = 1,500$

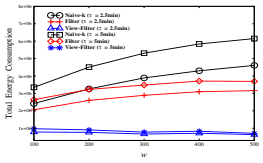


(c) The ratio of maximum energy consumption when $n = 1,000$

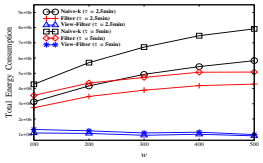


(d) The ratio of maximum energy consumption when $n = 1,500$

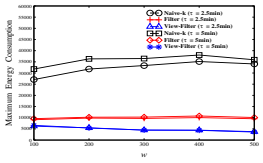
Fig. 2. The performance of algorithm `Filter` for time interval top-k query evaluation with $30 \leq k \leq 50$



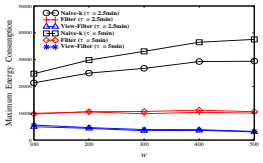
(a) The total energy consumption when $n = 1,000$



(b) The Total energy consumption when $n = 1,500$



(c) The maximum energy consumption when $n = 1,000$



(d) The maximum energy consumption when $n = 1,500$

Fig. 3. The performance of various algorithms for online time interval top-k queries for both network sizes $n = 1,000$ and $n = 1,500$ at time step $3,600$

B. Performance of Time Interval Top-k Query Evaluation

We first evaluate the performance of algorithm `Filter` for top-k query evaluation on a snapshot dataset with the range of k from 30 to 150. We assume that l is the number of points at each sensor, which is one of 1, 10, 20 and 50. The $l * n$ points in all sensors compose the snapshot dataset on which various algorithms are applied. Fig. 2(a)-(d) show the curves of the ratio of the total energy consumption and maximum energy consumption by algorithm `Filter` to those by algorithm `Naive-`k with different values of l for network sizes $n = 1,000$ and $n = 1,500$, respectively. It can be seen that the total energy consumption and the maximum energy consumption among the sensors by algorithm `Filter` are substantially less than those by algorithm `Naive-`k with various values of k and l. This shows that the proposed algorithm reduces unlikely top-k points in the network from transmission significantly. Moreover, algorithm `Filter` exhibits much better performance with the increase of l, which implies that it is highly scalable.

C. Performance of Online Time Interval Top-k Queries

We then study the performance of different algorithms for online interval top-k queries on real sensory datasets [9]. We assume that the duration of each time step is τ minutes in the simulation and the lifespan of all the points is w time steps. In our experiments, we evaluate the performance of the proposed algorithm by setting τ as 2.5 and 5 minutes respectively, and w ranges from 100 to 500 with an increment of 100. We generate a list of queries which represents that a top-$(k_t, [t_s, t_e])$ query is issued at time step t, where k_t ranges from 30 to 100, and t ranges from 1 to 3600. We evaluate the performance of algorithm `View-Filter` against that of algorithm `Filter` and algorithm `Naive-`k for each query with various values of τ and w. Fig. 3(a)-(d) show the curves of the total energy consumption and maximum energy consumption by various algorithms at time step $3,600$. It can be seen that algorithm `View-Filter` has the best performance among the mentioned algorithms in terms both metrics. Furthermore, the ratio of the total energy consumption and the maximum energy consumption by algorithm `View-Filter` to those by algorithm `Naive-`k becomes smaller with the increases of w and τ, which implies that algorithm `View-Filter` is highly scalable.

VI. Conclusions

In this paper we have tackled the top-k query processing in wireless sensor networks. We first advocated the concept of time interval top-k query. We then proposed an optimization framework for time interval top-k query evaluation, consisting of a filter-based algorithm on snapshot datasets, and algorithm `View-Filter` for online time interval top-k queries with various ks and time intervals under sliding window environments. The experimental results showed that the proposed algorithms are very efficient, and can prolong the network lifetime significantly.

References

[1] S. Madden, M. J. Franklin, J. M. Hellerstein, W. Hong. TAG: a tiny aggregation service for ad hoc sensor networks. *ACM SIGOPS Operating Systems Review*, Vol.36, pp.131–146, 2002.

[2] B. Babcock and C. Olston. Distributed top-k monitoring. *Proc. of ACM SIGMOD*, ACM, pp.28–39, 2003.

[3] M. Wu, J. Xu, X. Tang, W-C. Lee. Top-k monitoring in wireless sensor networks. *IEEE Trans. Knowledge and Data Engineering*, Vol.19, No.7, pp.962–976, 2007.

[4] R. Fagin, A. Lotem, M. Naor. Optimal aggregation algorithms for middleware. *J. Computer and System Science*, Vol.1, pp.614-656, 2001.

[5] G. Das, D. Gunopulos, N. Koudas, N. Sarkas. Ad-hoc top-k query answering for data streams. *Proc of VLDB*, pp183–194, 2007.

[6] D. Zeinalipour-Yaztia, Z. Vagenab, V. Kalogerakic, D. Gunopulosd, V. J. Tsotrase, M. Vlachosf, N. Koudas, D. Srivastavah. The threshold join algorithm for top-k queries in distributed sensor networks. *Proc of DMSN*, pp.61–66, 2005.

[7] G. J. Pottie, W. J. Kaiser Wireless Integrated Network Sensors. *Communication of ACM*, Vol.43 No.5, pp.51–58, 2000.

[8] Crossbow Inc. *MPR-Mote Processor Radio Board Users Manual.*

[9] `http://db.csail.mit.edu/labdata/labdata.html`

[10] Network Simulator-ns2. `http://www.isi.edu/nsnam/ns`, 2006.

[11] Y. Yao, J. Gehrke. The cougar approach to in-network query processing in sensor networks. *ACM SIGMOD Record*, Vol. 31, pp.9–18, 2002.

[12] W. Liang, B. Chen, J. X. Yu. Energy-efficient skyline query processing and maintenance in sensor networks. *Proc of ACM CIKM*, ACM, pp.1471–1472, 2008.

[13] A. Silberstein R. Braynard, C. Ellis, K. Munagala, J. Yang. A sampling-based approach to optimizing top-k queries in sensor networks. *Proc. of ICDE*, IEEE, pp.68–79, 2006.