driving, engine started, engine-only driving, engine driving/charging, compound driving, regenerative braking, mechanical braking, compound braking, and charging while standstill mode. A model of the EVT based SPHEV has been developed in the MATLAB/SIMULINK environment. A rule-based energy management control strategy using the efficiency maps of the components for estimating the power performance of the SPHEV has been presented.

The fuel consumption of the proposed SPHEV is 5.62% lower than that of the "Improved THS-like" vehicle, which indicates the fuel economy potential of this concept. One reason is that spin losses can be avoided by using the clutches.

The life cycle of the electric machines is expected to be effectively extended, because the open or lock operations of the clutches allow efficient usages in the proposed SPHEV. In addition, the speed of the electric machines is effectively confined to an acceptable range, compared with that in the "Improved THS-like" vehicle, which implies ease of manufacture, good sustainability, and low cost.

References

[1] C. C. Chan, "The state of the art of electric, hybrid, and fuel cell vehicles," *Proc. IEEE*, vol. 95, no. 4, pp. 704–718, Apr. 2007.

[2] M. J. Hoeijmakers, "The electric variable transmission," *IEEE Trans. Ind. Appl.*, vol. 42, no. 4, pp. 1092–1100, Jul./Aug. 2006.

[3] R. H. Staunton, C. W. Ayers, L. D. Marlino, J. N. Chiasson, and T. A. Burress, "Evaluation of 2004 Toyota Prius hybrid electric drive system," U.S. Dept. Energy Freedom CAR Vehicle Technol., Washington, DC, ORNL/TM-2006/423, 2006.

[4] H. Benford and M. Leising, "The lever analogy: A new tool in transmission analysis," presented at the Soc. Automotive Eng. Transaction, Warrendale, PA, 1981, SAE Paper 810102.

[5] M. Ehsani, Y. Gao, and J. M. Miller, "Hybrid electric vehicle: Architecture and motor drives," *Proc. IEEE*, vol. 95, no. 4, pp. 719–728, Apr. 2007.

[6] A. Takasaki, T. Mizutani, K. Kitagawa, T. Yamahana, K. Odaka, T. Kuzuya, Y. Mizuno, and Y. Nishikawa, "Development of new hybrid transmission for 2009 Prius," in *Proc. EVS24 Int. Battery, Hybrid Fuel Cell Elect. Veh. Symp.*, Stavanger, Norway, 2009.

[7] J. D. Wishart, Y. Zhou, and Z. Dong, "Review, modelling and simulation of two-mode hybrid vehicle architecture," in *Proc. ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, Las Vegas, NV, 2007.

[8] M. Cipek, J. Deur, and J. Petrić, "Bond graph modeling and analysis of series–parallel hybrid electric vehicle transmissions," presented at the Soc. Automotive Eng. World Congr. Exh., Detroit, MI, 2010, SAE Paper 2010-01-1309.

[9] M. R. Schmidt, "Two-mode, input-split, parallel, hybrid transmission," U.S. Patent 5 558 588, Sep. 24, 1996.

[10] A. Villeneuve, "Dual mode electric infinitely variable transmission," in *Proc. Int. CVT Hybrid Transm. Congr.*, Davis, CA, 2004.

[11] X. Ai, T. Mohr, and S. Anderson, "An electromechanical infinitely variable speed transmission," presented at the Soc. Automotive Eng. World Congr. Exh., Detroit, MI, 2004, SAE Paper 2004-01-0354.

[12] T. S. Birch and C. Rockwood, *Automatic Transmission & Transaxles*. Englewood Cliffs, NJ: Prentice-Hall, 2009.

[13] W. Xiong, Y. Zhang, and C. Yin, "Optimal energy management for a series–parallel hybrid electric bus," *Energy Convers. Manage.*, vol. 50, no. 7, pp. 1730–1738, Jul. 2009.

[14] Y. Zhang, H. Lin, B. Zhang, and C. Mi, "Performance modeling and optimization of a novel multi-mode hybrid powertrain," *Trans. ASME, J. Mech. Des.*, vol. 128, no. 1, pp. 79–89, Jan. 2006.

[15] J. Liu and H. Peng, "Modeling and control of a power-split hybrid vehicle," *IEEE Trans. Control Syst. Technol.*, vol. 16, no. 6, pp. 1242–1251, Nov. 2008.

[16] K.-B. Sheu, "Simulation for the analysis of a hybrid electric scooter powertrain," *Appl. Energy*, vol. 85, no. 7, pp. 589–606, Jul. 2008.

[17] [Online]. Available: http://en.wikipedia.org/wiki/Hybrid_Synergy_Drive

[18] O. Bitsche and G. Gutmann, "Systems for hybrid cars," *J. Power Sources*, vol. 127, no. 1/2, pp. 8–15, Mar. 2004.

[19] L. Chen, G. Xi, L. Luo, and C. Yin, "Improve engagement performance for automated clutch using model referenced adaptive controller," *J. Syst. Simul.*, vol. 21, no. 16, pp. 5102–5104, 2009.

On Efficient Processing of Continuous Historical Top-k Queries in Sensor Networks

Jie Cheng, Hongbo Jiang, *Member, IEEE*, Jiangchuan Liu, *Senior Member, IEEE*, Wenyu Liu, *Member, IEEE*, and Chonggang Wang, *Senior Member, IEEE*

***Abstract*—The top-k query has long been an important topic in computer science. Efficient implementation of top-k queries is the key for information searching. In this paper, we develop the Efficient algorithm for the Continuous Historical Top-k (ECHT) extraction, which is a novel algorithm that can effectively process the continuous historical top-k query. A simple top-k extraction algorithm based on aggregation is used for user query processing, and two additional steps on the filter setting by which individual nodes do not have to report all their readings are proposed to further reduce communication cost. To the best of our knowledge, this is the first work for continuous historical top-k query processing in sensor networks, and our simulation results show that our schemes can reduce the total communication cost by up to two orders of magnitude, as compared with the centralized scheme or a straightforward extension from the previous top-k algorithm on a continuous monitoring query.**

***Index Terms*—Algorithm/protocol design, sensor networks, top-k extraction.**

I. Introduction

As energy conservation is crucial to the prolonged lifetime of a sensor network [1], many approaches [2], [3] have been explored, with particular attention to aggregate query processing. Among all those aggregates, the top-k query is crucial for many applications such as environment monitoring, network management, and biology analysis. Previous studies strive to propose energy-efficient processing approaches for the top-k query such that the list of k highest (or lowest) sensor readings is retrieved.

For many applications, users may want to continuously extract or aggregate data such as top-k results from sensor networks for later analysis. For example, one question posed by the sink could be "what is the average speed of sensor nodes for detecting vehicle speed whose readings are the lowest top-k within the past 1 hour?" In this case, if the sink receives most of the top-100 results from the same node (or its neighbors), this area where this node is located is possibly suffering from traffic congestion. Despite its importance, unfortunately, none of previous works have been done on the *continuous historical top-k extraction* problem in sensor networks. The "monitoring" top-k query

Manuscript received August 30, 2010; revised January 2, 2011 and February 6, 2011; accepted April 18, 2011. Date of publication April 29, 2011; date of current version June 20, 2011. This work was supported in part by the National Natural Science Foundation of China under Grant 60803115, Grant 60873127, Grant 60903096, and Grant 61073147; by the National Natural Science Foundation of China and Microsoft Research Asia under Grant 60933012; by the Youth Chenguang Project of Wuhan City under Grant 201050231080; by the Scientific Research Foundation for the Returned Overseas Chinese Scholars (State Education Ministry); and by the Program for New Century Excellent Talents in University (State Education Ministry). The review of this paper was coordinated by Prof. A. Jamalipour.

J. Cheng, H. Jiang (Corresponding author), and W. Liu are with Department of Electronics and Information Engineering, Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: hongbojiang2004@gmail.com).

J. Liu is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada.

C. Wang is with InterDigital Communications, USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2011.2148203

has been well studied over the past few years [4]–[6]; it focuses on querying the top-k results in the readings at the current time epoch: the so-called *snapshot*. Although the snapshot top-k results could be extended to answer the top-k query in one or several time epochs, the "monitoring" top-k query may incur significant overhead. In contrast, the historical queries (i.e., queries over sliding windows such as in [7]) are of special interest in this paper. Those proposed top-k algorithms [4]–[6] in wireless sensor networks (WSNs) designed for continuous monitoring queries may be inefficient for processing continuous historical queries.

To address this problem, in this paper, we present a novel Efficient algorithm for the Continuous Historical Top-k (ECHT) extraction that can provide continuous top-k results used for the historical query while incurring communication cost in a scalable fashion. Apart from database communities or traditional distributed networks, we focus on the continuous top-k query of historical readings in WSNs. The sensor networks are discrete, and the top-k results have to be computed in a hop-by-hop fashion instead of the centralized fashion that is mostly used in the database community [7]. The characteristic of the wireless connection with multihop communication to the sink also differs our algorithm from those distributed top-k algorithms such as [8], where well-connected and wired connections are provided. In addition, as we have mentioned, the continuous historical queries are of special interest in this paper.

The remaining part of this paper proceeds as follows: Section II presents the ECHT algorithm and explores the reason why it can work. Section III reports on experiments. Section IV summarizes the main contribution of this paper and discusses several issues related to the proposed algorithm.

II. ECHT Algorithm

A. Overview

Here, we discuss how our proposed ECHT works. Initially, the sink collects all the sensor readings at first s epochs (as detailed in Section II-B). The sink sorts the sensor readings and obtains the initial top-k results set at first s time epochs. Accordingly, the sink estimates the initial filters across sensor nodes. Thereafter, at each sampling instance (from time epoch $s+1$), each sensor node sends an update to the sink, which is forwarded by intermediate nodes (see Section II-C). The sink will then probe the top-k results (see Section II-D) and reestimate the filters. It then sends the updated filters to relevant sensor nodes if necessary (see Section II-E). This process is performed at each sampling instance.

B. Estimation of the Ideal Filter at the Sink

We use the following confidence interval method to estimate the popularity of the top-k results conservatively. It is noted that we are not interested in the real data distribution of all readings. Instead, the number of data points that are larger than a certain real number f, i.e., the complementary cumulative distribution function (ccdf), is of special interest. Let the ccdf be $P(f)$, which represents the probability of one data point larger than f. By doing so, the data point search process is considered as a sequence of Bernoulli trials. Let S denote the current sample size. The probability of obtaining k results is given by the following binomial distribution: $\mathrm{Prob}(k|S) = \binom{S}{k} P(f)^k \cdot (1 - P(f))^{S-k}$, where $\binom{S}{k}$ denotes S chooses k. When sample size S is large, this probability approaches the following Poisson distribution: $(m^k/k!)e^{-m}$, where $m = P(f) \cdot S$. This Poisson distribution has mean m, variance m, and standard deviation $\sqrt{m}$. Then, we use Pearson's confidence interval on the Poisson distribution as follows: Given the number of top-k results k, we choose a conservative estimation m' of the true mean m by solving $m' - \delta\sqrt{m'} = k$, where the constant $\delta > 0$. We obtain $m' = k + \delta^2/2 + \delta\sqrt{k + \delta^2/4}$, which is the upper limit of Pearson's confidence interval. This result says that, if there are $k > 0$ top-k results, then the expected number of returned results is less than m' with a probability determined by the parameter δ. For example, if $\delta = 2$, the confidence level is about 95%, and the corresponding f' is denoted by $f'_{\delta=2}$ according to $m' = P(f') \cdot S$.

Notably, the aforementioned approach implies that if we try to study historical top-k queries, the filter should be updated, and the ECHT algorithm has a conservative estimation of $m' = k + \delta^2/2 + \delta\sqrt{k + \delta^2/4}$. This leads to two different cases. T denotes the set of current data points collected by the sink, and $|T| = l$ denotes the number of members in $T = \{d_{n_1}, d_{n_2}, \ldots, d_{n_l}\}$. First, when the sink obtains enough l $(l \geq m')$ data points, the current filter is reset to be $f' = d_{n_{m'}} \cdot v$ such that there are m' data points not less than f'. Second, the received results are insufficient (i.e., $l < m'$). In this case, we assume that the data follow a uniform distribution between f' and $d_{n_l} \cdot v$ when the probability density function is $P(x) = (d_{n_1} \cdot v - d_{n_l} \cdot v)/l$ for $f' \leq x \leq d_{n_l} \cdot v$. Accordingly, the filter is reset to be $f' = d_{n_l} \cdot v - (m' - l) \cdot P(x)$. By doing so, the number of data points larger than f' should be larger than or equal to k with the probability of 95% for $\delta = 2$, where $f'_{\delta=2}$ denotes the corresponding f' with $\delta = 2$. Roughly speaking, for a top-100 extraction, if we strive to retrieve the first $122 (m' = 100 + 4/2 + 2\sqrt{100+1} \approx 122)$ largest value of sensor readings by setting the filter, often, at least 100 readings in the current time slot will be forwarded with the probability of 95%.

C. Updating and Hop-by-Hop Aggregation

Since an ordinary sensor node could be far away from the sink, it would incur significant communication cost if the node sends readings to the sink via a multihop fashion. To address this problem, in-network aggregation techniques are exploited in ECHT. The sink, which is based on the relationship between the node readings and its filters, adjusts the filter setting faithfully. As a result, WSNs can benefit from aggregation to save messages at the same time. Each node maintains a set of history data points based on the previously generated local top-k results that are sent to its parent at the last time epoch. When receiving top-k candidates from its children, the node aggregates these local top-k results and sends updates to its parent.

Initially, each node sends updates to its parent from leaf nodes. Each leaf node sends updates only when the readings (not marked as "transported") belong to its k largest readings, which are greater than filter f_i. After nonleaf nodes receive all their children's updates, they sort the readings received from their children with the generated readings and find the k biggest reading set U_i. Then, they send all the readings (not marked as "transported") received from their children directly to their parent regardless of their filter. In addition, each node sends its generated readings, which are in the local top-k results and beyond its filter as well. That is, a data point is sent to its parent when $d_i \cdot v > f_i$ and $d_i \cdot v \in U_i$. Finally, each sensor node needs to mark the forwarded readings as "transported," whether or not the readings are generated or received from its children, to avoid redundant transmission.

D. Probe Algorithm

Although we have carefully designed filter f_{prev} as aforementioned, insufficient top-k values are still possible, i.e., $|T_0| < k$, where T_0 is the current data set collected by the sink, which is all beyond f_{prev}. Thus, the sink will have to probe some nodes to requery the top-k

results: the so-called "probe" process. Obviously, there are some readings filtered out by filters, as mentioned in Section II-C; thus, the sink could send a value p_0 less than f_{prev} to each node whose filter f_i is higher than p_0. After these nodes receive p_0, they report all their readings that are higher than p_0 and have not been sent to the sink before. Then, the sink will receive all the readings beyond p_0. T_1 denotes the current reading set collected by the sink, readings in which are all beyond p_0. If the number of members in reading set T_1 is still less than k, the sink has to probe again with a value p_1 less than p_0 until the sink obtains at least k readings all beyond certain p_J, where J is a positive integer. For the similar reason mentioned in Section II-B, we could choose a suitable value $f'_{\delta=2,T=T_0}$, which denotes the estimation of $f'_{\delta=2}$ with a confidence level of 95% for $T = T_0$ and $p = p_0$. Since we choose $p_0 = f'_{\delta=2,T=T_0}$ to probe, there is a very high probability of getting enough readings. Finally, we will get enough readings beyond $p_J = f'_{\delta=2,T=T_J}$ successfully. In this case, obviously, $p_i < p_j \ \forall J \geq i > j \geq 0$ according to Section II-B, and the probability of obtaining sufficient top-k results is $1 - (1 - 0.95)^j$, $j = 1, 2, \ldots, J$ (which converges to 1 quickly).

E. Filter Setup at Nodes

It is noted that, in our framework, each node holds its local filter f_i, instead of the global filter f calculated by the sink. This is due to the fact that it is costly to frequently notify all nodes to update the filter. Our idea here is to classify these filters into several (or a few) classes to reduce communication cost via multicasting techniques. To that end, we intensively classify the nodes whose filters need to be adjusted into four sets, which are denoted by S_0, S_1, S_2, and S_3, in which the nodes should update their local filter accordingly. The classification of the nodes depends on whether their readings have contributed to top-k results recently.

Often, the nodes related to global top-k values are only a fraction of all nodes at a certain epoch. This motivates that we only adjust the filters of these nodes that may affect top-k results, avoiding adjusting the filters of other nodes. Thus, we can define the node set in which nodes may affect top-k results, depending on their reported readings, which are denoted by H, and its previous time epoch version H_{prev} in Algorithm 1 (lines 3–4, where cur is the current time epoch).

Algorithm 1 NodeClassification()

```
1:Initialize sets $S_0$, $S_1$, $S_2$, $S_3$ to empty set
2:for every node $i$ do
3:   if $d_i \cdot v \geq f'_{\delta=1}$ where $cur - d_i.t < s$ then
4:      $H \leftarrow H \cup \{i\}$; // $cur$ is the current time epoch
5:   else
6:      if $i \in H_{\text{prev}}$ then
7:         $S_1 \leftarrow S_1 \cup \{i\}$;
8:      end if
9:   end if
10:end for
11:if $f'_{\delta=3} \leq f_{\text{prev}} \leq f'_{\delta=1}$ then
12:   $f \leftarrow f_{\text{prev}}$; // do not update the filter
13:   $S_0 \leftarrow H - H_{\text{prev}}$; // only for those nodes requiring
      update
14:else
15:   $f = f'_{\delta=2}$; // update the filter
16:   $S_0 \leftarrow H$;
17:   if $f_{\text{prev}} > f'_{\delta=1}$ then
18:      for node $i$ such that $f_i > f \wedge i \notin S_0 \cup S_1$ do
19:         $S_2 \leftarrow S_2 \cup \{i\}$
20:      end for
21:   end if
22:end if
23:for node $i \notin S_0 \cup S_1 \cup S_2$ do
24:   if $d_i \cdot v > f_i$ where $d_i.t == cur$ then
25:      $S_3 \leftarrow S_3 \cup \{i\}$
26:   end if
27:end for
```

Since the sensor networks are normally temporally and spatially correlated [9], the variation of H slowly changes. Thus, sets S_0 (while f does not update), S_1, and S_2 usually have quite a few elements. Because updating of f is very slow, as aforementioned, filter updating of S_0 (while f updates) is not frequent. Despite that the nodes in S_3 are by reason of normal growth or outlier, a larger filter can prevent frequent updating of the nodes for a certain long time. Thus, the number of nodes in $S_0 \cup S_1 \cup S_2 \cup S_3$ is very small (even none). As such, the traffic caused by updating filters can be significantly reduced.

Algorithm 2 shows the filter updates according to the node classification in Algorithm 1. First, as the nodes in S_0 are constantly contributing to top-k results, their filters will have to update in accordance with the estimation of ideal filter f at the sink (lines 4–6). Second, for those nodes in $S_1 \bigcup S_2$ that stopped contributing to the top-k results in the current epoch shown in Algorithm 1 (lines 6–8 and 18–20), we have intensively updated a more conservative filter (lines 7–9) such that these nodes cannot be classified into the set of H recently. Here, the goal is to avoid updating the filter frequently at these nodes. Third, for those nodes in S_3, they probably report top-k results due to normal growth or outlier. In this case, we do not increase the local filter f_i of node i too much (lines 10–12), but we only set it between the reported readings $d_i \cdot v$ and $f'_{\delta=3}$. Finally, the sink only informs those nodes in $S_0 \cup S_1 \cup S_2 \cup S_3$ (often a few nodes) to update the local filters.

Algorithm 2 FilterSetting()

```
1:if $S_0 \cup S_1 \cup S_2 \cup S_3 == \emptyset$ then
2:   Return;
3:end if
4:for node $i \in S_0$ do
5:   $f_i \leftarrow f$;
6:end for
7:for node $i \in S_1 \cup S_2$ do
8:   $f_i \leftarrow f'_{\delta=3}$;
9:end for
10:for node $i \in S_3$ do
11:   $f_i \leftarrow (d_i \cdot v + f'_{\delta=3})/2$;
12:end for
13:Notify all nodes in $S_0 \cup S_1 \cup S_2 \cup S_3$ to update the local
   filter $f_i$
```

III. Performance Evaluation

To evaluate the performance of our proposed framework, we conduct a series of experiments. We have implemented the simulator, and in this paper, we quantify several performance metrics of our algorithms. We first describe our data sets and the alternative techniques and then present the results and analysis.

A. Data Sets

Intel Berkeley Laboratory Data: This publicly available sensor data set was collected by the Intel Berkeley Research Laboratory for 1 mo [10]. The data consist of environmental data regularly collected from 54 nodes spread around their laboratory. We observed some missing data values for various nodes at different time epochs and filled there in with the average of the values during the previous and subsequent epochs at the same node. In our simulation, we select the entire temperature records for one week (February 28–March 5). We choose a node close to the center of the area as the sink in each experiment.

Synthetic data: To evaluate the algorithms in larger networks and with larger data sets, we also generated synthetic networks and synthetic data. The size of synthetic networks ranges from 100 to 1000 nodes. We have used a random placement of nodes with a uniform probability distribution. Each node has, on average, five nodes within its radio range. The data at every node i are modeled as $x_t = \alpha_i x_{t-1} + e_t$, where $e_t \sim N(0, 0.1)$ and $\alpha_i \sim N(1, 0.01)$. For each node, 2880 data values were generated. Every node is initialized with $\alpha_1 = 1$ and $x_0 = 0$. This model is updated in every measurement.

B. Alternative Techniques

First, we implement "centralized exact" [11] and "FILA-S" [4] to show that these methods are inefficient for our scenario. Second, as mentioned in Section I, there are no algorithms designed for the scenario in this paper. In comparison with our idea, we study "FILA-S" and "TAG" to design "FILA-A" and "Basic scheme without diffusion," respectively, under the historical top-k query scenario. Third, since the necessary top-k results need to be forwarded to the sink, regardless of the aforementioned approaches, we calculate the necessary traffic cost as the "optimal case" to show how the total traffic cost and lifetime are close to minimal and maximum, respectively.

C. Evaluation Results

Here, to answer the query presented in Section I, our simulation strives to retrieve *the k largest values across sensor readings within the past 1 h.* We evaluate the algorithms' performance in terms of the total power consumption of transmitted and received packets by all sensor nodes. In addition to the necessary updating traffic to the sink, the total traffic costs also include unnecessary (abundant) updating and overhead for probing and modifying filters.

Fig. 1 shows the total number of transmitted packets using the Intel Berkeley Laboratory data on the first day as a function of k. First, the ECHT algorithm obviously outperforms others, and it reduces the total traffic significantly. It only causes less than 10% traffic compared with the centralized algorithm and about 30% traffic compared with a simple extension of the previous FILA-S algorithm for a modest k (e.g., not more than 200). Second, ECHT only introduces 27% traffic compared with the centralized algorithm, even when $k = 1000$ when the minimal total traffic is 19% of the centralized algorithm. Third, all the three algorithms based on the idea of using aggregation (including the basic scheme without diffusion, FILA-A, and ECHT) are scalable. Fourth, when k is large, for example, more than 600, the performance improvement seems inconspicuous. The reason is that, with a 1-h time slot, only around 6000 sensor readings are

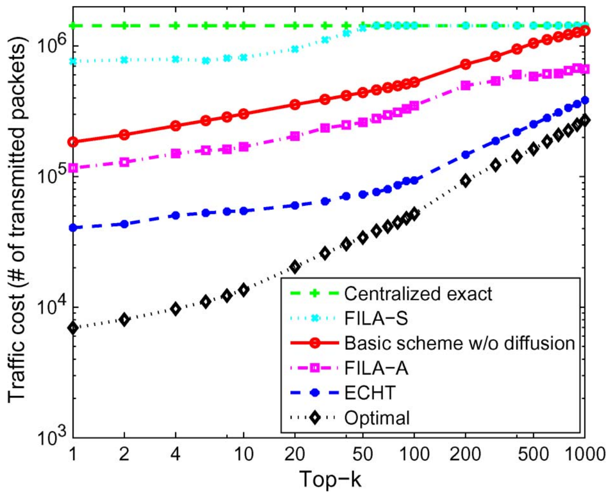

Fig. 1. Traffic comparison using the Intel Berkeley Laboratory data with varying k values.

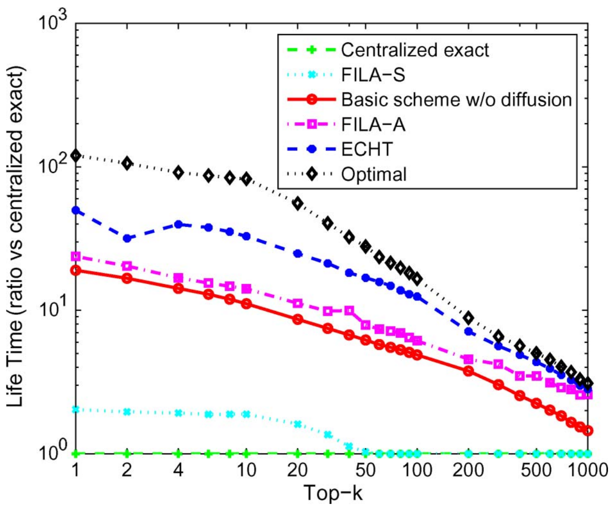

Fig. 2. Lifetime comparison using the Intel Berkeley Laboratory data with varying k values.

generated, which is our data set to perform the top-k query. Finally, from this figure, we conclude that the previous FILA-S algorithm is only suitable for the case of monitoring queries. ECHT, which targets historical query processing, exploits the aggregation and dynamic filter settings so that it is able to achieve much better performance.

The lifetime of sensor networks is defined by the time duration of the first sensor node that runs out of its power. Thus, we find the maximum power consumption $M_{\max}$ by one node in all nodes on the first day using each aforementioned method under different k. Then, we denote the maximum power consumption by one node in all nodes using "centralized exact" as M_0, and it is obviously the same under different k. Finally, we define $(1/M_{\max})/(1/M_0)$ as the appraisal value of lifetime. Fig. 2 demonstrates the effectiveness of ECHT on prolonging the lifetime of sensor networks compared with other approaches. First, ECHT outperforms others several to dozens times in most instances. Second, ECHT extremely prolongs the lifetime of WSNs. For example, ECHT achieves 12.5 times lifetime versus the "centralized exact" scheme when $k = 100$. Even under an ultimate instance, e.g., $k = 1000$, ECHT can prolong the lifetime of WSNs about three times compared with the "centralized exact"

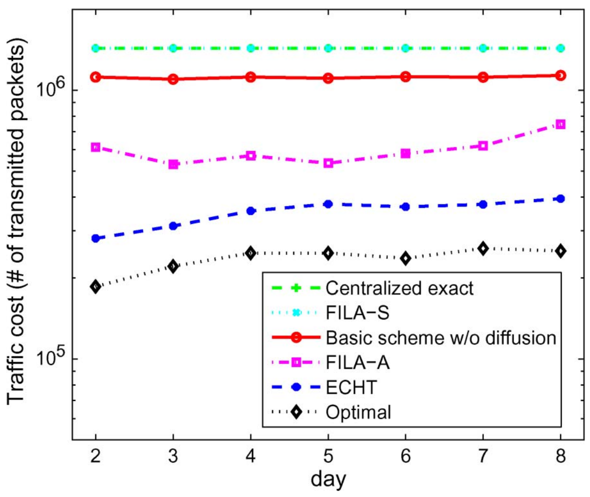

Fig. 3. Comparison using the Intel Berkeley Laboratory data over a one-week period.

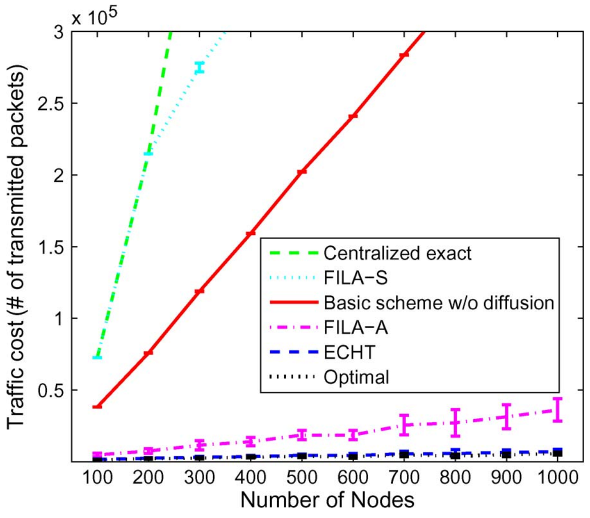

Fig. 4. Comparison with the synthetic data set with a variety of network sizes.

scheme. Third, the performance of ECHT is very close to the "optimal" case. When $k < 30$, the "optimal" case is only two to three times the lifetime of ECHT. Once $k \geq 30$, the "optimal" case is not more than two times the lifetime of ECHT and even only 1.09 times with $k = 1000$.

In our simulation, we also fix k and study the stability of our algorithms over time. Fig. 3 depicts the communication cost of answering top-200 queries, which is top 10%, over a one-week period on the Intel data with different algorithms. Again, our algorithms outperform others, including the "centralized exact" algorithm and FILA-S. Specifically, the ECHT algorithm, as compared with the "centralized exact" algorithm, where all nodes continuously send back their data, achieves much better performance with only 19%–27% as much communication cost.

Fig. 4 compares the different algorithms with the synthetic data for answering top-200 queries for the data set with a range of network sizes. For each point, we run the simulation ten times and present the average results. Obviously, the communication cost using ECHT only linearly increases with the number of sensor nodes. In contrast, the cost increases at noticeably higher rates with other algorithms. As the network grows in size, ECHT shows a very marginal increase in communication costs (for example, less than 1% for 1000 nodes) compared with the "centralized exact" algorithm.

IV. Conclusion

In this paper, we have proposed ECHT, which is a novel algorithm that efficiently processes the continuous historical top-k queries in WSNs. To reduce the communication cost, we first proposed the necessaries of the filter. Then, to accurately estimate an appropriate filter based on previous information, we utilized the confidence interval method with Pearson's confidence interval to conservatively estimate the popularity of the top-k results. An interesting finding is that we do not have the knowledge of the exact data distribution itself by considering the data point search process as a sequence of Bernoulli trials. Instead, the only requirement is that the data distribution does not significantly change over time, as we have mentioned. It is noted that we do not set a uniform filter across all sensor nodes. Instead, one basic idea behind our algorithm is that we strive to update the filter at nodes as few as possible. That is, those nodes not contributing to the top-k results often keep a constant filter with no change over time. For those nodes contributing to the top-k results, the filter is more conservative to retrieve enough results, as well as to avoid the frequent filter update. Moreover, we found an aggregation technique that is helpful to reduce the incurred traffic for historical top-k query processing as well. We finally use extensive simulation results to demonstrate the efficiency of our proposed ECHT.

Some extensions can be considered. First, ECHT offers an accurate estimation for the filter for top-k queries such that an approximate top-k monitoring extension is straightforward. We are able to choose different estimations of the appropriate filter based on the probability with which the enough k largest value could be forwarded to the sink. Second, we plan to do real testing in real sensor networks. Third, we plan to utilize a multipath routing scheme to achieve robustness when link/node failures are considered.

References

[1] S. He, J. Chen, Y. Sun, D. K. Y. Yau, and N. K. Yip, "On optimal information capture by energy-constrained mobile sensors," *IEEE Trans. Veh. Technol.*, vol. 59, no. 5, pp. 2472–2484, Jun. 2010.

[2] N. Shrivastava, C. Buragohain, and D. Agrawal, "Medians and beyond: New aggregation techniques for sensor networks," in *Proc. ACM SenSys*, 2004, pp. 239–249.

[3] H. Jiang, S. Jin, and C. Wang, "Parameter-based data aggregation for statistical information extraction in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 8, pp. 3992–4001, Oct. 2010.

[4] M. Wu, J. Xu, X. Tang, and W.-C. Lee, "Top-k monitoring in wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 7, pp. 962–976, Jul. 2007.

[5] D. Zeinalipour-Yazti, Z. Vagena, D. Gunopulos, V. Kalogeraki, V. Tsotras, M. Vlachos, N. Koudas, and D. Srivastava, "The threshold join algorithm for top-k queries in distributed sensor networks," in *Proc. DMSN*, 2005, pp. 61–66.

[6] A. Silberstein, K. Munagala, and J. Yang, "Energy-efficient monitoring of extreme values in sensor networks," in *Proc. ACM SIGMOD*, 2006, pp. 169–180.

[7] K. Mouratidis, S. Bakiras, and D. Papadias, "Continuous monitoring of top-k queries over sliding windows," in *Proc. ACM SIGMOD*, 2006, pp. 635–646.

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

[9] M. Vuran, Ö. Akan, and I. Akyildiz, "Spatio-temporal correlation: Theory and applications for wireless sensor networks," *Comput. Netw.*, vol. 45, no. 3, pp. 245–259, Jun. 2004.

[10] [Online]. Available: http://db.lcs.mit.edu/labdata/labdata.html

[11] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: An acquisitional query processing system for sensor networks," *ACM Trans. Database Syst.*, vol. 30, no. 1, pp. 122–173, Mar. 2005.