Information Sciences 181 (2011) 869–882

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Top-*k* query evaluation in sensor networks under query response time constraint

Weifa Liang [a,*], Baichen Chen [a], Jeffrey Xu Yu [b]

[a] School of Computer Science, Australian National University, Canberra, ACT 0200, Australia
[b] Department of Systems Engineering and Engineering Management, Chinese University of Hong Kong, Shatin, NT, Hong Kong

ARTICLE INFO

Article history:
Received 17 June 2008
Received in revised form 14 May 2009
Accepted 8 October 2010

Keywords:
Top-*k* queries
Wireless sensor network
QoS constraint
Routing trees
Query optimization
Energy conservation
Query response time
Network lifetime

ABSTRACT

Top-*k* query in a wireless sensor network is to find the *k* sensor nodes with the highest sensing values. To evaluate the top-*k* query in such an energy-constrained network poses great challenges, due to the unique characteristics imposed on its sensors. Existing solutions for top-*k* query in the literature mainly focused on energy efficiency but little attention has been paid to the query response time and its effect on the network lifetime. In this paper we address the query response time and its effect on the network lifetime through the study of the top-*k* query problem in sensor networks with the response time constraint. We aim at finding an energy-efficient routing tree and evaluating top-*k* queries on the tree such that the network lifetime is significantly prolonged, provided that the query response time constraint is met too. To do so, we first present a cost model of energy consumption for answering top-*k* queries and introduce the query response time definition. We then propose a novel joint query optimization framework, which consists of finding a routing tree in the network and devising a filter-based evaluation algorithm for top-*k* query evaluation on the tree. We finally conduct extensive experiments by simulation to evaluate the performance of the proposed algorithms, in terms of the total energy consumption, the maximum energy consumption among nodes, the query response time, and the network lifetime. The experimental results showed that there is a non-trivial tradeoff between the query response time and the network lifetime, and the joint query optimization framework can prolong the network lifetime significantly under a specified query response time constraint.

1. Introduction

Technological advances in recent years have enabled the deployment of large-scalable sensor networks consisting of hundreds or thousands of inexpensive sensors in an ad hoc fashion for a variety of environmental monitoring and surveillance applications including measurement of meteorological data (e.g. temperature, pressure, humidity), noise levels, chemicals, etc. [1,17]. During the course, a large volume of sensed data generated by sensors is needed to be aggregated within the network, and the result is needed to be relayed to the base station to respond to users queries. The sensor network thus is treated as a *virtual database* by the database community [14]. This type of database, however, is essentially different from the traditional database. For example, it is impossible to store all sensed data in a central site (the base station) due to the large volume of continuing sensing data generated by sensors. On the other hand, the battery-powered sensors will quickly become inoperative due to the large quantity energy consumption if all sensed data were sent to the base station for storage

* Corresponding author. Tel.: +61 2 6125 3019; fax: +61 2 6125 0010.
E-mail address: wliang@cs.anu.edu.au (W. Liang).

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.10.006

or processing. Furthermore, the lifetime of the network is closely tied to energy consumption rates at the sensors. Hence, how to query the network effectively and efficiently is an important and challenging issue. In particular, radio communication (transmission and reception) is the primary source of energy consumption in sensor networks [19], minimizing radio communication in query execution can save large amounts of energy and thereby prolonging the network lifetime significantly. Several studies on different query optimization in sensor networks have been conducted recently [21,10,2].

Query optimization in traditional databases has been extensively studied, and typical optimization metrics are the query response time and space required. Unlike their wired counterparts, query optimization in wireless sensor networks focused on energy efficiency in order to prolong the network lifetime. Since energy conservation has dominated most of the research in energy-constrained sensor networks, the concepts of query response time, end-to-end delay and jitters have not been taken into account in most published works. However, along with the introduction of imaging and video sensors, the increasing interest in many real-time applications of wireless sensor networks, including disaster management, combat field and security surveillance, has posed additional challenges. That is, in these applications our optimization objective is not only to prolong network lifetime but also to meet the certain end-to-end delay constraint (e.g. the query response time).

Top-k query is one of very popular queries in wireless sensor networks, which is to find the k nodes with the highest values (readings) among the sensor nodes. One such an example is a sensor network deployed for monitoring the air pollution index of a region of interest. A typical top-k query issued to the network is to find the k sensors with the highest pollution index readings at any given time point. Although several studies on top-k query evaluation and maintenance in sensor networks have been conducted recently [23–25], none of them takes the query response time into consideration while focusing on energy efficiency only. However, in some real-time applications, the query response time is very critical. For example, consider a sensor network used to detect the forest fires. When a forest fire happens, the forest management authority may issue a top-k query to request the vicinity images of the k sensors with the highest temperature readings. Such a query has a stringent query response time imposed, because timing is very crucial for fire fighters to distinguish the fires. The faster the system responds to the query, the sooner the fire fighters are able to figure out the whereabout of fires accurately and timely.

1.1. Related work

Top-k query evaluation in distributed environments has been extensively studied in the literature [7,3,4,15,29]. Typically, there are two types of top-k queries. One is the distributed top-k query which aims to find the k highest ranked objects, where the ranking score of an object is an aggregated value from a number of attribute values stored at distributed sources. Many algorithms for this type of top-k query have been developed, which includes the Threshold Algorithm (TA) [7], the Three-Phase Uniform Threshold algorithm (TPUT) [4], the KLEE algorithm [15] and the distributed Threshold Join Algorithm (TJA) [29]. Another is to find the k nodes with the highest readings in a sensor network, assuming that each node generates a sensing reading. For this latter one, Wu et al. [24,25] exploited the semantics of top-k query and proposed a novel Filter-based algorithm (FILA) for monitoring the top-k results. Formally speaking, their algorithm maintains the top-k readings continuously by assigning a specific interval of key values for each sensor node, according to the current top-k key values, and this interval serves as the filter for the node to suppress its unnecessary updates. Meanwhile, the base station keeps a copy of the filter of each sensor node to maintain an (error bound) approximate view of the node's reading. A sensor node reports its updated reading to the base station only when the reading passes its filter. Otherwise, the updated reading will be ignored. Consequently, the total energy consumption for top-k query evaluation can be reduced significantly through the reduction of unnecessary data transmission within the network. The energy savings delivered by their solution however is based on the assumption that each sensor node is within the transmission range of the base station, each updating probe broadcast by the base station can be heard by all sensor nodes, and the total reception energy consumption of all sensor nodes was not taken into account. In real life, this assumption is too restrictive, the total reception energy consumption of all sensor nodes by receiving the base station's probing message cannot be ignored, because it is well known that the reception energy consumption of a sensor node is about one third of its transmission energy consumption in most short-distance wireless communications. For example, the reception energy consumption on MICA2 mote is 14.4 mJ/s, while its transmission energy consumption is only 36 mJ/s [6,23]. Therefore, if the sensing readings among sensor nodes are updated frequently, the probing cost will become prohibitively expensive. Silberstein et al. [23] considered the top-k query problem in sensor networks by providing several approximate solutions with high probability, based on top-k sampling of the past readings. They demonstrated the power and flexibility of sampling-based approach by formulating the problem under an energy constraint as linear programming and developed a series of top-k query planning algorithms, assuming that the sensor network is a tree network.

A closely related problem that considered both the query response time and the network lifetime is the data gathering problem subject to the end-to-end delay constraint. In fact, the top-k query with the response time constraint is a special case of this general setting. Despite that there are several studies on data gathering that tradeoff the end-to-end delay and the network lifetime [20,11,28], they are either inapplicable or have their limitations on this special case, because they assume that either the message length is fixed and given in prior or the data transmission rate at each node is dynamically adjustable. For example, Lindsey et al. [11] proposed an optimization metric $energy \times delay$ for data gathering and demonstrated that a chain-based routing tree delivers the longest network lifetime. However, finding such a chain in sensor

networks is intractable, instead, a heuristic algorithm is proposed. Prabhakar et al. [20] observed that, in many channel coding schemes, the transmission energy at each relay node can be significantly reduced by lowering the transmission power and increasing the duration of transmission. Techniques such as modulation scaling [22] have been proposed for implementing such tradeoffs. Yu et al. [28] considered data gathering by providing an optimal schedule for packets on the tree network to meet the end-to-end delay constraint. They also devised an offline centralized optimal algorithm with the assumption that the delay and packet length are given in prior, and on-line approximation algorithm based on dynamic programming to meet the delay constraint through adjusting the transmission rate of individual nodes on the routes.

1.2. Contributions

In this paper we study the top-k query problem with response time constraint in wireless sensor networks. We first address the query response time and its effect on the network lifetime explicitly. We then propose a novel joint query optimization framework that consists of finding a routing tree in the network and devising an algorithm for top-k query evaluation on the tree in an efficient manner to meet the specified response time constraint. We finally conduct extensive experiments by simulations to evaluate the performance of the proposed optimization framework, in terms of the total energy consumption, the maximum energy consumption among the nodes, the query response time, and the network lifetime. The experimental results show that there is a non-trivial tradeoff between the query response time and the network lifetime, and the proposed joint optimization framework can prolong the network lifetime significantly under the given query response time constraint.

1.3. Paper organization

The remainder of the paper is organized as follows. In Section 2, the sensor network model and the problem definition are introduced. An existing algorithm for answering top-k queries without any query response time constraint is briefly introduced. In Section 3, the cost model of energy consumption and the time delay by transmitting a message is proposed, followed by introducing the energy consumption model for answering top-k queries and the query response time. In Section 4, a simple joint query optimization framework is proposed, which consists of finding a routing tree and applying an existing algorithm `NAIVE`-k for top-k query evaluation on the tree such that the network lifetime is maximized. In Section 5, an efficient joint query optimization framework is proposed, which incorporates filtering at nodes into the design of the evaluation algorithm to filter out unnecessary data within the network from transmission. In Section 6, extensive experiments by simulations are conducted to evaluate the performance of the proposed algorithms. The conclusions are given in Section 7.

2. Preliminaries

2.1. System model

We consider a sensor network consisting of n sensor nodes randomly deployed in a region of interest, each measuring a numeric value (reading). Assume that there is a base station with unlimited energy supply, which serves as a gateway between the sensor network and users. That is, the users issue queries to and get answers from the network through the base station. On the other hand, the battery-powered sensor nodes are responsible for sensing and collecting sensing data from their vicinities. They are also capable of processing the sensed data and transmitting the aggregate data to their neighbors, where two sensor nodes are *neighbors* if they are within the transmission range of each other. Each sensor node has identical transmission range. Let R mJ and r_e mJ be the amounts of energy consumed of a sensor node by transmitting and receiving a one-byte data, respectively. A sensor network can be represented by an undirected graph $G(V,E)$, where V is the set of sensor nodes and the base station, E is the set of links. There is a link in E between two sensor nodes if they are within the transmission range of each other. We assume that a single communication channel is shared by all sensor nodes in the network.

2.2. Problem definition

Given a sensor network $G(V,E)$, assume that each sensor node has a sensing reading val_i, a top-k query is to find the k nodes with the highest readings in the network, $1 \leqslant i, k \leqslant n$. *The top-k query problem with response time constraint* in G is defined as follows. Given a top-k query issued at the base station in a wireless sensor network $G(V,E)$ and a specified query response time bound Γ, the problem is to evaluate the query in-network by presenting an evaluation plan such that the network lifetime is maximized, provided that the query response time constraint Γ is met as well, assuming that a routing tree rooted at the base station is used for the query evaluation, where the *network lifetime* is referred to as the failure time of the first node in the network [5]. The rationale behind the network lifetime definition is the imbalance of energy consumption among the sensor nodes. The sensor nodes near to the base station consume much more energy than the others by relaying messages for them, thus they often die first. Once they are dead, the base station will be disconnected from the rest of sensor nodes in the network no matter how much residual energy left and how well connected the rest of sensor nodes.

2.3. An existing algorithm

We briefly review an algorithm NAIVE-k for top-k query evaluation without response time constraint, proposed by Silberstein et al. [23] for later use.

Algorithm NAIVE-k computes the answer in a bottom-up fashion by one pass over the network. Each node simply collects the top-k results from each of its children, computes the top-k among all such readings and its own, and passes them to its parent. Specifically, a leaf node just forwards its reading to its parent. For an internal node, if the node contains $k'(\leqslant k)$ readings (including its own one), it forwards all the readings to its parent. Otherwise, it forwards its top-k readings to its parent. In the end, the root identifies the top-k readings from all collected readings from its children and its own, which are the top-k result of the top-k query.

Algorithm NAIVE-k aims to reduce the energy overhead per message by minimizing the number of messages transmitted. It actually achieved the minimum number of messages transmitted, since each node must be visited at least once in order to return the top-k exact results for a top-k query. It however has the maximum number of readings transmitted. For example, if a node has d_v children, then its reception energy consumption is $d_v(l_H + k * l) * r_e$ in the worst case, assume that each child holds the top-k readings, where l is the number of bytes to represent a reading and l_H is the equivalent number of bytes needed of the transmission energy consumption for the overhead on the message header and handshaking. If both k and d_v are relatively large, node v will die quickly due to its energy expiration. Thus, to prolong the network lifetime by devising energy-efficient algorithms for top-k query evaluation, there is a non-trivial tradeoff between the number of messages transmitted and the number of readings contained in each message during the evaluation of the query.

It is worthwhile to mention another evaluation algorithm for top-k query evaluation, referred to as the pipeline algorithm NAIVE-1 by Silberstein et al. [23], which transmits the minimum number of readings but a large number of messages within the network. Experimental results in [23] showed that it is beaten by algorithm NAIVE-k when a network consists of 200 nodes with $k \geqslant 4$. In terms of the response time, the pipeline algorithm has a much longer time delay, since each time each parent requests next top readings from its children in a pipeline fashion, which obviously cannot meet the stringent query response time constraint in some real-time applications. Thus, we will use algorithm NAIVE-k for comparison purpose.

3. Energy consumption and the response time for top-k queries

3.1. Energy consumption and time delay of a message transfer

In wireless sensor networks, we assume that the basic communication mechanism is messages. Two neighboring nodes can send messages to each other. A message consists of a header and a message body, where the header usually contains the source node and destination node addresses and the other fault-tolerant information, which is represented by l_h bytes, whereas the message body contains from zero to multiple sensing readings. *A reading* of a node is composed of the ID of the node and the sensing value, which is represented by l bytes. If a message contains k readings, then its message length is $l_h + l * k$. If the message is transmitted from node u to node v within one hop, then the transmission energy consumption at node u is $(l_h + l * k) * R$, while the reception energy consumption at v is $(l_h + l * k) * r_e$. The total energy consumption associated with this message transmission is $(l_h + l * k) * (R + r_e)$, where $l_h(R + r_e)$ is the energy overhead on the message header and identical for all messages, while the value $l * k * (R + r_e)$ of a different message is different, depending on how many readings the message body contains. Intuitively, the energy consumption on transmitting the message header is significantly smaller than that on transmitting the number of readings contained in the message body. However, in practice, to build a reliable communication channel between two neighboring nodes, a handshaking between them is needed before proceeding message transmissions. The energy consumption on handshaking and the message header transmission is actually much higher than that on just transmitting the message header. We thus refer to *the energy overhead per message* as the energy overhead on both handshaking and the message header transmission. The energy overhead per message is significantly higher than that on transmitting the message body containing just a single value. This can be witnessed from a commercial sensor product MICA2 mote [6,23]. It costs only 0.02016 mJ/byte by transmitting a byte of data but consumes around 0.645 mJ in handshaking and the message header transmission [23], which are equivalent to the amounts of energy needed to transmit 32 bytes of data.

To measure the energy consumption of a message under the transmission energy measure, we convert the energy overhead per message into an equivalent amounts of transmission energy consumed by transmitting ρ readings into a formula $\mathcal{E}_H = (\rho * l) * \mathcal{E}_{byte}$, where $\mathcal{E}_{byte}$ is the energy consumption on transmitting a single byte data and $\mathcal{E}_H$ is the energy overhead per message. For the sake of convenience, we can convert the energy overhead per message into the equivalent amounts of transmission energy on transmitting l_H bytes data, then $\rho = l_H/l$.

The time delay incurred by a message transmission is dominated by the *transmission time* as there is no queuing delay. The processing and propagation delays are negligible compared to the transmission time delay. Denote by t_H the time spent per message consisting of the time spent on handshaking and message header transmission,which is identical for all messages. The transmission time for a message body is various, which is proportional to the length of message body. If a message body contains k readings, then its transmission time is kl/s, assuming that all sensor nodes have an identical, fixed data transmission rate s. As a result, the transmission time of a message containing k readings is $t_H + kl/s$.

3.2. Energy consumption of answering a top-k query

Following the convention in TAG [13], the energy-efficient top-k query evaluation in sensor networks can be implemented through in-network processing paradigm [13,16,26,27]. Specifically, a routing tree $\mathcal{T}$ rooted at the base station and spanning all sensor nodes will be used for query evaluation. Let $parent(u)$ denote the parent of node u and $C(v)$ the child set of v in $\mathcal{T}$. The query evaluation on $\mathcal{T}$ consists of two phases: the *distribution stage* in which the query is pushed down to each node along the tree paths, and the *data collection stage* where the sensed data of sensor nodes are routed up from children to parents and eventually are routed to the base station through multiple-hop relay.

To answer a top-k query in an energy-aware manner, an evaluation algorithm for the query that maximizes the network lifetime is desperately needed, in terms of a specific energy optimization metric. It is well known that the typical energy optimization objective is to minimize either the total energy consumption or the maximum energy consumption among the nodes, where the total energy consumption objective can be achieved by reducing the network traffic through minimizing not only the number of messages transmitted but also the number of readings contained in each message, whereas the maximum energy consumption objective can be achieved by minimizing the energy consumption among the maximum degree nodes in $\mathcal{T}$ to prolong the lifetime of these nodes, thereby prolonging the network lifetime. Specifically, suppose that there is an algorithm $\mathcal{A}$ for top-k query evaluation on a tree $\mathcal{T}$ that involves M messages transmission within the tree. Assume that N is the number of readings contained by the M messages. Then, the total energy consumption for answering this top-k query by algorithm $\mathcal{A}$ on tree $\mathcal{T}$ is $\mathcal{E}_{total}(\mathcal{A},\mathcal{T}) = (Ml_H + Nl)(R + r_e)$. Meanwhile, for a given node v in $\mathcal{T}$ of d_v children, we assume that the number of messages sent or received by v is $M^{send}(v)$ or $M^{rece}(v)$, and the number of readings contained by the corresponding messages is $N^{send}(v)$ or $N^{rece}(v)$, respectively. Then, the total energy consumption of v for this query is $\mathcal{E}(v) = (M^{send}(v)l_H + N^{send}(v)l)R + (M^{rece}(v)l_H + N^{rece}(v)l)r_e$. Thus, the maximum energy consumption among the nodes by algorithm $\mathcal{A}$ on tree $\mathcal{T}$ is $\mathcal{E}_{max}(\mathcal{A},\mathcal{T}) = \max_{v\in V}\{\mathcal{E}(v)\}$.

To prolong the network lifetime when answering top-k queries, it is required to find a routing tree $\mathcal{T}$ and to devise an algorithm $\mathcal{A}$ on $\mathcal{T}$ such that both $\mathcal{E}_{total}(\mathcal{A},\mathcal{T})$ and $\mathcal{E}_{max}(\mathcal{A},\mathcal{T})$ are minimized. However, finding such a tree and devising such an algorithm are difficult, since their objectives are conflicting with each other. We thus focus on finding a feasible routing tree and an evaluation algorithm that strikes the right balance between these two-conflicting optimization objectives.

3.3. The response time to answer a top-k query

Given a routing tree $\mathcal{T}$ rooted at the base station r, assume that a node v has d_v children $u_1, u_2, \ldots$ and u_{d_v}. Recall that a single communication channel is used in the network, which implies that each time only one child can communicate with its parent. We further assume that an algorithm $\mathcal{A}$ for top-k query evaluation on $\mathcal{T}$ in the bottom up fashion. That is, each leaf node sends its result to its parent. An internal node v in $\mathcal{T}$ starts sending its top-k readings to its parent only if (i) it received all necessary readings (data) from its children and there is not any further transmission from its children; and (ii) its children can communicate with the node as many rounds as needed before the node starts communicating with its parent. Thus, the time delay at node v is defined as the moment it gets all necessary data from all its children and just starts its own data transmission to its parent. More precisely, *the response time* $t(r)$ to answer a top-k query by applying an algorithm $\mathcal{A}$ on a routing tree $\mathcal{T}$ rooted at base station r can be defined recursively as follows.

Let $t(u)$ be the time delay at node u. Assume that the d_v children of node v, $u_1, u_2, \ldots, u_{d_v}$, are indexed in increasing order of time delay, i.e., $t(u_1) \leqslant t(u_2) \leqslant \cdots \leqslant t(u_{d_v})$. Then, the earliest time that v received all the top-k readings from its children is

$$t(v) = \begin{cases} 0, v \text{ is a leaf}, \\ \max\limits_{1\leqslant i\leqslant d_v}\left\{\left(t(u_i) + \sum\limits_{j=i}^{d_v} tr(u_j, M_j)\right) + tp(v)\right\}, \end{cases} \quad (1)$$

where $tr(u_j, M_j)$ is the transmission time to transmit M_j messages from u_j to v, and each message contains no more than k readings, $1 \leqslant j \leqslant d_v$. $tp(v)$ is the processing time at v to identify the top-k readings in the subtree rooted at v after receiving the readings from its children, which is proportional to the number of readings received. Obviously, $tp(v)$ is negligible compared with the transmission delay, we thus set $tp(v) = 0$. The response time to a top-k query thus is $t(r)$.

4. A simple joint optimization framework

In this section we propose a simple joint optimization framework for top-k query evaluation, which takes into account both the energy optimization and the response time constraint simultaneously. That is, we aim to find a routing tree such that the network lifetime derived by applying algorithm NAIVE-k on the tree is maximized. Meanwhile, the response time constraint is met as well.

4.1. Query response time vs. network lifetime

Unlike previous studies that assume the existence of an available routing tree and focused on how to evaluate a query by data aggregation, data compression, and varying data transmission rate at nodes, we not only focus on the energy

optimization for query evaluation but also consider the construction of the routing tree that affects the query response time. We propose an approach which exhibits a great flexibility to find a specific routing tree for top-k query evaluation. As a sensor network contains many different routing trees, we can easily find such a specific tree for our purpose.

To motivate our discussion, we consider two extreme routing trees for top-k query evaluation, one is the maximum degree-constrained spanning tree T_{MDST} in which the maximum node degree is minimized, and another is the Breadth-First-Search tree T_{BFS}. Let Δ_{MDST} and Δ_{BFS} be the maximum node degrees of T_{MDST} and T_{BFS}, respectively. We examine the relationship between the query response time and the network lifetime when applying an evaluation algorithm, e.g. algorithm NAIVE-k, on T_{MDST} and T_{BFS} as follows.

In terms of query response time, the depth of T_{MDST} is much deeper than that of T_{BFS}, while the depth of T_{BFS} is the lowest among all spanning trees rooted at the base station. This implies that the response time to a top-k query by applying an algorithm on T_{BFS} usually is much shorter than that by applying the same algorithm on T_{MDST}, because the former explores more parallelism of message transmissions among the nodes, while the latter is inherently sequential in the worst scenario. On the other hand, the maximum energy consumption among the nodes on T_{BFS} is significantly larger than that on T_{MDST}, because the maximum energy consumption among the nodes in a tree is proportional to its maximum node degree and the number of descendants of each of its children. Since $\Delta_{MDST} < \Delta_{BFS}$, this implies that the network lifetime delivered by T_{MDST} is substantially longer than that by T_{BFS}, provided that the same evaluation algorithm is applied to both of them. As there is a non-trivial tradeoff between the query response time and the network lifetime, it is crucial to find an appropriate routing tree for top-k query evaluation that can not only prolong the network lifetime significantly but also meet the specified query response time constraint. Such a tree should have the following properties.

For given a node v with less than k descendants, the sensing readings of all descendants of node v will be relayed to v eventually. The total energy consumption of relaying a descendant's reading to v is proportional to the distance (the number of hops) between the descendant and v, the shorter the distance between them, the less amounts of energy it consumes. On the other hand, for a node with no less than k descendants, although its maximum transmission energy consumption is fixed (due to the fact that it transmits its top-k readings to its parent), its reception energy consumption is various, depending on how many children it has and how many top readings contained by each of its children. The more children it has, the more reception energy it consumes. To minimize the reception energy consumption of node v, it requires that the number of children d_v of v be as small as possible. In addition, a maximum degree node v that received kd_v children readings will consume the maximum amounts of energy compared with the other nodes in the tree. Thus, to minimize the maximum energy consumption among the nodes in a routing tree, the maximum node degree in the tree must be minimized. The *maximum degree-constrained spanning tree* (MDST for short) is a nice candidate, which is a spanning tree rooted at the base station that the maximum node degree is minimized. It has been shown that finding a MDST in a graph is NP-complete [9]. Instead, there is a nearly optimal approximation algorithm [8], which delivers a routing tree in which the maximum node degree is $\Delta^* + 1$, where Δ^* is the maximum node degree in the optimal solution. Motivated by that, in the following we propose a simple joint optimization framework for top-k query evaluation.

4.2. Algorithm for finding routing trees

We use algorithm NAIVE-k as the evaluation algorithm and T_{MDST} as the initial routing tree, which has the minimum maximum node degree among all the spanning trees rooted at the base station r. This implies that applying algorithm NAIVE-k on T_{MDST} will deliver a much longer network lifetime. The response time to a top-k query obtained by using this tree however is prohibitively slow due to the fact that it is a skinny tree with long depth. To meet the response time constraint, the proposed strategy aims to meet the constraint gradually at the expense of a small fraction of network lifetime. That is, we modify the current routing tree by increasing the maximum node degree of these nodes with no less than k descendants by one and updating the descendants of a node into its children if possible. As a result, the depth of the resulting tree is lowered, while the maximum energy consumption among the nodes in the resulting tree increases. The network lifetime thus becomes shorter. This procedure continues until the resulting routing tree meets the response time constraint.

Specifically, let UE be the upper bound of the maximum energy consumption among the nodes in the current routing tree. Let Δ be the maximum degree of nodes with no less than k descendants in the tree. It can be seen that a node with maximum degree Δ will consume the maximum amounts of energy among the tree nodes if each child of the node contains the top-k readings of nodes in the subtree rooted at the child. Then, $UE = (l_H + kl)R + \Delta(l_H + kl)r_e = (\rho + k)lR + \Delta(\rho + k)lr_e$, where the term $(l_H + kl)R$ is the transmission energy consumption of transmitting the top-k readings to its parent and the term $\Delta(l_H + kl)r_e$ is its reception energy consumption of receiving all top-k readings from its children.

Suppose that the current routing tree does not meet the query response time constraint, we modify the tree iteratively until it meets the response time constraint. To do so, we distinguish the nodes in the current routing tree by two cases. Case (i) a node v with less than k descendants. If a descendant of v is within its transmission range and the total energy consumption at v by adding the descendant as its child is not greater than UE, the descendant becomes a child of v in the resulting routing tree. Case (ii) a node v with no less than k descendants. If the grandparent of v is within its transmission range and the grandparent degree is strictly less than Δ, node v sets itself as a child of its grandparent in the resulting routing tree. The simple joint query optimization of finding an energy-efficient routing tree $\mathcal{T}$ and applying algorithm NAIVE-k for top-k query evaluation on $\mathcal{T}$ is described as follows:

Algorithm. `Find_Routing_Tree_k` (G, k, r, Γ, `NAIVE-k`)

begin
1. find an approximate MDST T_{app} in G using algorithm in [8],
   let Δ_{MDST} be the maximum degree of nodes in T_{app}.
2. compute the number of descendants $desc(v)$ of each node v in T_{app};
3. $t(r) \leftarrow \infty$; $\mathcal{T} \leftarrow T_{app}$; /* $t(r)$ is the query response time */
4. $\Delta \leftarrow \Delta_{MDST} - 1$; $\Delta(G) \leftarrow$ maximum node degree of G;
5. **while** ($t(r) > \Gamma$) and ($\Delta < \Delta(G)$) **do**
6. $\Delta \leftarrow \Delta + 1$; /* increasing the max node degree by 1 */
7. $UE \leftarrow (\rho + k)lR + \Delta(\rho + k)lr_e$;
8. **for** each $v \in V$ **do**
9. **if** $desc(v) < k$
10. **then** $grandp(v) \leftarrow parent(parent(v))$;
11. **while** ($desc(grandp(v)) < k$) and ((v,$grandp(v)$) $\in E$) **do**
12. $E(grandp(v)) \leftarrow (\rho + desc(grandp(v)))lR + desc(grandp(v))lr_e$
    $+ d_{grandp(v)}\rho r_e + \rho r_e$; /* $E(v)$ is the energy consumption at node v */
13. **if** $E(grandp(v)) \leqslant UE$
14. **then** $C(parent(v)) \leftarrow C(parent(v)) - \{v\}$;
15. $desc(parent(v)) \leftarrow desc(parent(v)) - desc(v)$;
16. $d_{parent(v)} \leftarrow d_{parent(v)} - 1$;
17. $parent(v) \leftarrow grandp(v)$;
18. $C(grandp(v)) \leftarrow C(grandp(v)) \cup \{v\}$;
19. $d_{grandp(v)} \leftarrow d_{grandp(v)} + 1$;
    /* d_v is the degree of v in the current routing tree*/
20. **else**
    while ($d_{grandp(v)} < \Delta$) and ((v,$grandp(v)$) $\in E$) **do**
21. $C(parent(v)) \leftarrow C(parent(v)) - \{v\}$;
22. $desc(parent(v)) \leftarrow desc(parent(v)) - desc(v)$;
23. $d_{parent(v)} \leftarrow d_{parent(v)} - 1$;
24. $parent(v) \leftarrow grandp(v)$;
25. $C(grandp(v)) \leftarrow C(grandp(v)) \cup \{v\}$;
26. $d_{grandp(v)} \leftarrow d_{grandp(v)} + 1$;
27. compute $|S(v)|$ of each node v in the resulting routing tree;
    /* $S(v)$ is the set of readings received from the children of v */
28. $tr(r) \leftarrow$ `Compute_Response_Time` ($\mathcal{T}, k, r$);
end.

The response time estimate is given by the following subroutine `Compute_Response_Time`.

Function `Compute_Response_Time` ($\mathcal{T}, k, v$):*real*
begin
1. **if** node v is a leaf node in $\mathcal{T}$ **then return** 0;
   /* let $u_1, u_2, \ldots, u_{d_v}$ be the children of v */
2. **else for** $j \leftarrow 1$ **to** d_v **do**
3. $t(u_j) \leftarrow$ `Compute_Response_Time`($\mathcal{T}, k, u_j$);
4. sort $t(u_i)$ in increasing order, let $u_{i_1}, u_{i_2}, \ldots, u_{i_{d_v}}$ be the ordered list of nodes in time delay,
5. $t(v) \leftarrow -\infty$; /* the maximum time delay at v */
6. **for** $t \leftarrow 1$ **to** d_v **do**
7. **if** $t(u_{i_t}) + \sum_{j=t}^{d_v}(t_H + \min\{k, l_{i_j}\}l/s) > t(v)$
   then $t(v) \leftarrow t(u_{i_t}) + \sum_{j=t}^{d_v}(t_H + \min\{k, l_{i_j}\}l/s)$;
8. **return** $t(v)$;
end.

Having the routing tree $\mathcal{T}$, applying algorithm `NAIVE-k` for top-k query evaluation on $\mathcal{T}$ is straightforward, and the actual response time is no greater than the constraint Γ. The network lifetime delivered by this approach is significantly prolonged in comparison with that delivered by applying algorithm NAIVE-k on T_{BFS}, which will be verified by later experimental simulations.

5. An efficient joint query optimization framework

In this section we propose an energy-efficient joint query optimization framework for top-k query evaluation subject to a response time constraint, which consists of finding a routing tree and devising a filter-based evaluation algorithm for top-k query evaluation on the found tree. The filters installed at individual nodes aim to filter out unnecessary data from transmission within the network to reduce data traffic in the network, thereby reducing the transmission energy consumption within the network.

5.1. Top-k query filtering algorithm

Suppose that each child u_i of node v holds the top-l_i readings of nodes in the subtree of $\mathcal{T}$ rooted at u_i, let $L(u_i) = \{s_{i,1}, s_{i,2}, \ldots, s_{i,l_i}\}$ be the set of the top-l_i readings and the elements in $L(u_i)$ have been sorted in decreasing order, $s_{i,j} \geqslant s_{i,j'}$ if $j < j'$, $1 \leqslant i,j,j' \leqslant d_v$, $3 \leqslant l_i \leqslant k$. Clearly, the top-$k$ readings of nodes in the subtree rooted at v are in set $S(v) = \bigcup_{u_i \in C(v)} L(u_i)$ and v's own reading. The cardinality of $S(v)$ can be computed as follows. Assume that the number of descendants $desc(v)$ of v in $\mathcal{T}$ is given, where v is not its own descendant. Then (i) if $desc(v) < k$, $|S(v)| = desc(v)$; (ii) $|S(v)| = \sum_{i=1}^{d_v} \min\{k, desc(u_i) \mid u_i \text{ is a child of } v\}$, otherwise.

The intuition behind the filter-based algorithm is as follows. Assume that $|S(v)| \geqslant 2k$, find an element x in $S(v)$ such that $|R(x)| \geqslant k$, where $R(x) = \{a | a \in S(v) \text{ and } a \geqslant x\}$. The element x will be used as the filter of v to partition the elements in $S(v)$ into two disjoint subsets $S_x = \{b | b < x \text{ and } b \in S(v)\}$ and $S'(v) = S(v) - S_x$. Obviously, $R(x) \subseteq S'(v)$. Now, all the readings in S_x will be filtered out from transmission. Specifically, the proposed filtering algorithm is described below.

Each child u_i first sends the median $s_{i,\lceil l_i/2 \rceil}$ of its top-l_i readings to v, $1 \leqslant i \leqslant d_v$. Having received the median sequence consisting of d_v median readings, node v then sorts the sequence in decreasing order. Let $m_{i_1}, m_{i_2}, \ldots, m_{i_{d_v}}$ be the sorted sequence, $1 \leqslant i_j, j \leqslant d_v$. A filter $x = m_{i_j}$ is chosen such that j is the smallest integer in $|R(x)| = \sum_{p=1}^{j} \lceil l_{i_p}/2 \rceil \geqslant k$ with $1 \leqslant j \leqslant d_v$. Notice that such a j must exist due to $|S(v)| \geqslant 2k$. Third, node v broadcasts x to all its children. Each child u_i finds a smallest reading $s_{i,l_i'}$ in $L(u_i)$ such that $s_{i,l_i'} \geqslant x$ and transmits its top-l_i' readings except its median to v, $l_i' \leqslant l_i$. Node v finally identifies the top-k readings of nodes in the subtree rooted at itself, based on the received readings from its children and its own reading. For convenience, we refer to this algorithm as `FilterA`, which has the following properties.

Lemma 1. *If there is a node v in $\mathcal{T}$ with $|S(v)| \geqslant 2k$, then there must be a filter $x \in S(v)$ from the median sequence of its children such that $|R(x)| \geqslant k$. Let $R'(x)$ consist of the elements in the lower half of $L(u_{i_t})$ of node u_{i_t} that are not in $R(x)$ with $1 \leqslant t < j$, assuming that $x = m_{i_j}$. Then, $|R'(x)| \leqslant k - 1$ and $|S'(v)| \leqslant |S(v)|/2 + d_v + |R'(x)|$.*

Proof. Let $m_{i_1}, m_{i_2}, \ldots, m_{i_{d_v}}$ be the sorted median sequence and $x = m_{i_j}$. We claim the existence of such a j that $\sum_{t=1}^{d_v} \lceil l_{i_t}/2 \rceil \geqslant k$, $1 \leqslant j \leqslant d_v$. Otherwise, $\sum_{t=1}^{d_v} \lceil l_{i_t}/2 \rceil < k$. However, given $|S(v)| \geqslant 2k$, we have $\sum_{t=1}^{d_v} \lceil l_{i_t}/2 \rceil \geqslant \sum_{t=1}^{d_v} l_{i_t}/2 = |S(v)|/2 \geqslant k$. This results in a contradiction.

Following the definition of x, j is the minimum integer such that $\sum_{t=1}^{j} \lceil l_{i_t}/2 \rceil \geqslant k$, we have $\sum_{t=1}^{j-1} \lceil l_{i_t}/2 \rceil \leqslant k - 1$. Consequently, $|R'(x)| = \sum_{t=1}^{j-1} \lfloor l_{i_t}/2 \rfloor \leqslant \sum_{t=1}^{j-1} \lceil l_{i_t}/2 \rceil < k \leqslant k - 1$. Following algorithm `FilterA`, for a child u_{i_t} with $t < j$, it transmits no more than its top-l_{i_t} readings except $s_{i_t,\lceil l_{i_t}/2 \rceil}$ to v, for the child indexed with i_j, it transmits its top-$\lfloor l_{i_j}/2 \rfloor$ readings except x to v, while for a child u_{i_t} with $t > j$, it transmits no more than its top-$\lfloor l_{i_t}/2 \rfloor$ readings except $s_{i_t,\lceil l_{i_t}/2 \rceil}$ to v. Then, $|S'(v)| \leqslant \sum_{t=1}^{j-1} l_{i_t} + \lceil l_{i_j}/2 \rceil + \sum_{t=j+1}^{d_v} \lceil l_{i_t}/2 \rceil \leqslant \sum_{t=1}^{d_v} \lceil l_{i_t}/2 \rceil + |R'(x)| \leqslant |S(v)|/2 + d_v + k - 1$. □

In the end, the top-k readings in $S(v)$ are in $S'(v) = S(v) - S_x$, since $|R(x)| \geqslant k$, $R(x) \subseteq S'(v)$, and every element in S_x is no greater than any element in $R(x)$. Accordingly, almost a half of elements in $S(v)$ are filtered out from transmission, with a small extra cost by transmitting the d_v median readings of its children to v and broadcasting x to its children, when x is used as the filter of v.

5.2. Finding a routing tree with response time constraint

We now analyze the energy consumption and time delay at node v with and without filtering. Assume that the time delay $t(u_i)$ at u_i is given already, $1 \leqslant i \leqslant d_v$. Without loss of generality, we assume that the sequence of child nodes of v, $u_1, u_2, \ldots, u_{d_v}$, is sorted in increasing order of time delays at the nodes, i.e., $t(u_1) \leqslant t(u_2) \leqslant \cdots \leqslant t(u_{d_v})$.

If no filtering is applied to node v, then the total energy consumption of v by receiving all readings in $S(v)$ from its children is $E_{nofilt}(v) = (d_v l_H + |S(v)|l)r_e$, the time delay at v is

$$t_{nofilt}(v) = \max_{1 \leqslant i \leqslant d_v} \left\{ t(u_i) + \sum_{j=i}^{d_v} (t_H + \min\{k, l_j\} l/s) \right\}, \tag{2}$$

where child u_j contains the top-l_j readings of nodes in the subtree rooted at u_j, $1 \leqslant j \leqslant d_v$. Otherwise (a filter is applied to v), the total energy consumption of v by receiving the readings in $S'(v)$ and filtering is $E_{filt}(v) = 2d_v l_H r_e + (|S(v)|/2 + d_v + |R'(x)|)lr_e + (l_H + l)R$, and the time delay at v is

$$t_{filt}(v) = \max_{1 \leqslant i \leqslant d_v} \{ t(u_i) + (d_v - i + 1)(t_H + l/s) + t_H + (d_v t_H + (|S(v)|/2 + d_v + k)l/s) \}, \tag{3}$$

since $|S'(v)| \leqslant |S(v)|/2 + d_v + |R'(x)| \leqslant |S(v)|/2 + d_v + k$ by Lemma 1. Denote by $\delta E(v)$ and $\delta t(v)$ the differences of energy consumption and time delay between no filtering and filtering at v, when receiving the readings in $S(v)$ and $S'(v)$ respectively, then $\delta E(v) = E_{nofilt}(v) - E_{filt}(v) = (|S(v)|/2 - d_v(\rho + 1) - |R'(x)| - (\rho + 1)R/r_e)lr_e$ and $\delta t(v) = t_{nofilt}(v) - t_{filt}(v)$.

To ensure that filtering at node v results in energy savings, $\delta E(v) \geqslant 0$. Consequently, $|S(v)| \geqslant 2d_v(\rho + 1) + 2(k - 1) + 2(\rho + 1)R/r_e$. We state this by the following lemma.

Lemma 2. *Given a node v of d_v children in $\mathcal{T}$ and each child u_i contains top-l_i readings with $1 \leqslant i \leqslant d_v$, assume that $|S(v)| \geqslant 2k$. Then, filtering at v will guarantee energy savings if $|S(v)| \geqslant \theta(v)$, where $\theta(v) = 2d_v(\rho + 1) + 2(k - 1) + 2(\rho + 1)R/r_e$.*

Although Lemma 2 implies that filtering at v guarantees the energy savings if $|S(v)| \geqslant \theta(v)$, it does not mean that it guarantees the time delay reduction at v too. Therefore, filtering at a node is only performed when this will result in both energy saving and the time delay reduction.

Following the above analysis, an improved algorithm `Find_Routing_Tree_Filter` for finding a routing tree meeting the response time constraint is derived, through a minor modification to algorithm `Find_Routing_Tree`, which is described as follows: Replace the subroutine `Compute_Response_Time` based on algorithm `NAIVE-`k in line 28 by the following subroutine `Joint_Opt_Response_Time` based on the proposed filtering algorithm `FilterA`.

Function `Joint_Opt_Response_Time` $(\mathcal{T}, k, v)$:*real*
begin
1. **if** node v is a leaf node in $\mathcal{T}$ **then return** 0;
   /* let $u_1, u_2, \ldots, u_{d_v}$ be the children of v */
2. **else** compute $|S(v)|$;
3. $\theta(v) \leftarrow 2d_v(\rho + 1) + 2(k - 1) + 2(\rho + 1)R/r_e$;
4. **for** $j \leftarrow 1$ **to** d_v **do**
5. $t(u_j) \leftarrow$ `Joint_Opt_Response_Time`$(\mathcal{T}, k, u_j)$;
6. sort $t(u_i)$ in increasing order,
   let $u_{i_1}, \ldots, u_{i_{d_v}}$ be the nodes in time delay order.
7. $t_{nofilt}(v) \leftarrow -\infty$; $t_{filt}(v) \leftarrow -\infty$;
8. **for** $t \leftarrow 1$ **to** d_v **do**
9. **if** $t(u_{i_t}) + \sum_{j=t}^{d_v}(t_H + \min\{k, l_{i_j}\}l/s) > t_{nofilt}(v)$
   then $t_{nofilt}(v) \leftarrow t(u_{i_t}) + \sum_{j=t}^{d_v}(t_H + \min\{k, l_{i_j}\}l/s)$;
10. **for** $t \leftarrow 1$ **to** d_v **do**
11. $t_{temp}(v) \leftarrow t(u_{i_t}) + (d_v - t + 1)(t_H + l/s) + (d_v + 1)t_H + (|S(v)|/2 + d_v + k)l/s)$;
12. **if** $t_{temp}(v) > t_{filt}(v)$
    then $t_{filt}(v) \leftarrow t_{temp}(v)$;
13. $\delta t(v) \leftarrow t_{nofilt}(v) - t_{filt}(v)$;
14. **if** $(|S(v)| \geqslant \theta(v))$ and $(\delta t(v) \geqslant 0)$/* case (i) */
    then *filter*$(v) \leftarrow 1$; $t(v) \leftarrow t_{filt}(v)$; /* install a filter */
15. **else** *filter*$(v) \leftarrow 0$; $t(v) \leftarrow t_{nofilt}(v)$; /* no filter is installed at v */
16. **return** $t(v)$;
end.

As a result, a routing tree $\mathcal{T}$ has been built, which meets the specified query response time constraint.

5.3. Query evaluation algorithm with response time constraint

Having built the routing tree $\mathcal{T}$, algorithm `FilterA` for top-k query subject to the response time constraint is proposed, which consists of three subroutines `Initialization`, `Child_Node` and `Parent_Node` as follows:

Procedure `Initialization`(G, r, k)

begin
1. construct $\mathcal{T}$ in G by call `Find_Routing_Tree_Filter`$(\mathcal{T}, k, r, \Gamma,$ `FilterA`$)$;
2. compute the numbers of descendants *desc*(v) and children d_v of each node v;
3. each child u of node v sends its *desc*(u) value to v;
4. each parent node v broadcasts *desc*(u) of each child u, d_v, and *filter*(v) to all its children.
end.

Note that the initialization is only done once. Upon its execution, each node u_i in $\mathcal{T}$ will know the number of siblings d_v it has, the number of descendants *desc*(u_j) of each sibling u_j with $i \neq j$, and whether a filter *filter*$(parent(u_i))$ is installed at its parent v, $1 \leqslant i, j \leqslant d_v$. Each node in $\mathcal{T}$ then acts properly, according to the role (parent or child node) it plays by performing the following pseudo-code.

Procedure `Child_Node` $(\mathcal{T}, u_i, k)$

begin
1. **if** *filter*(*parent*(u_i)) = 0 **then** send its top-l_i readings to node v;
2. **else** send the median $s_{i,\lceil l_i/2 \rceil}$ of its top-l_i readings to v
3. and wait for a filter broadcast x from it.
4. identify the smallest element $s_{i,l_i'}$ in $L(u_i)$ such that $s_{i,l_i'} \geqslant x$ and transmit its top-l_i' elements in $L(u_i)$ to v;

end.

Procedure `Parent_Node` $(\mathcal{T}, v, k)$
begin
1. **if** *filter*(v) = 0 /* no filtering at its parent v */ **then** wait for receiving all readings in $S(v)$;
2. **else** wait for receiving the medians from its children
3. and sort the median sequence in decreasing order.
   /* Let $m_{i_1}, m_{i_2}, \ldots, m_{i_{d_v}}$ be the sorted sequence. */
4. filter $x = m_{i_j}$ is chosen such that $|R(x)| = \sum_{p=1}^{j} \lceil l_{i_p}/2 \rceil \geqslant k$, where j is the smallest integer with $1 \leqslant j \leqslant d_v$.
5. broadcast x to and receive data from its children;
6. identify the top-k readings of nodes in the subtree rooted at v, based on the received readings and its own reading;

end.

Theorem 1. *Given a top-k query in a wireless sensor network G(V,E) with the response time constraint Γ, let $\mathcal{T}$ be the resulting tree obtained by algorithm* `Find_Routing_Tree_Filter`. *Then, the network lifetime delivered by applying algorithm* `FilterA` *for top-k query evaluation on $\mathcal{T}$ is significantly prolonged in comparison with the one delivered by applying algorithm NAIVE-k on T_{BFS}. Meanwhile, the actual query response time meets the constraint Γ.*

Proof. Following the construction of the routing tree $\mathcal{T}$, it can be easily seen that the actual query response time by applying algorithm `FilterA` on $\mathcal{T}$ is no greater than Γ. In fact, the performance of Algorithm `FilterA` is at least as good as algorithm `NAIVE-k`, which is explained in the following.

During the construction of the routing tree $\mathcal{T}$, the estimation on the query response time is very conservative, since the time delay at a node v depends on not only the number of readings received from its children but also the time delay at each of its children, while the number of readings transmitted by all children under the filtering of x is no more than $|S(v)|/2 + d_v + |R'(x)|$ if the filter x is applied to v, where $1 \leqslant |R'(x)| \leqslant k-1$. In the construction of $\mathcal{T}$, $|R'(x)|$ is set to be $k-1$, which is very conservative. The actual number of readings transmitted from its children thus is much less than $|S(v)|/2 + d_v + |R'(x)|$, because many readings whose values are below the threshold x are filtered out from transmission. Thus, the actual time delay at v is no greater than the amounts estimated. □

6. Performance evaluation

In this section we evaluate the performance of the proposed algorithms in sensor networks, in terms of the total energy consumption, the maximum energy consumption among the nodes, the query response time, and the network lifetime through experimental simulation.

6.1. Simulation environment

We assume that the wireless sensor network is used to monitor a 100 m × 100 m region of interest. Within the region, 500 sensor nodes are randomly deployed by the NS-2 simulator [18], and the base station is located at the square center. The wireless sensor network $G(V,E)$ consisting of sensor nodes and the base station is then formed, where V is the set of sensor nodes and the base station. There is a bi-directional link in E between two nodes if they are within the transmission range of each other. We further assume that all sensor nodes have the same transmission range (10 meters in this paper) and G is connected. Although it is well known that sensing and the local computation at a sensor node also consume a certain amounts of energy, we here only take into account its energy consumption on radio communication, i.e., its transmission and reception energy consumptions. In all our experiments, we adopt the actual reception and transmission energy consumption parameters of MICA2 mote, where the amounts of reception energy and transmission energy are r_e = 0.00576 mJ and R = 0.0144 mJ per byte, respectively, and the transmission rate is 2500 bytes per second. For the time delay of transmitting a message, the time delay is 0.4 ms by transmitting one byte data, and t_H is 0.8 ms.

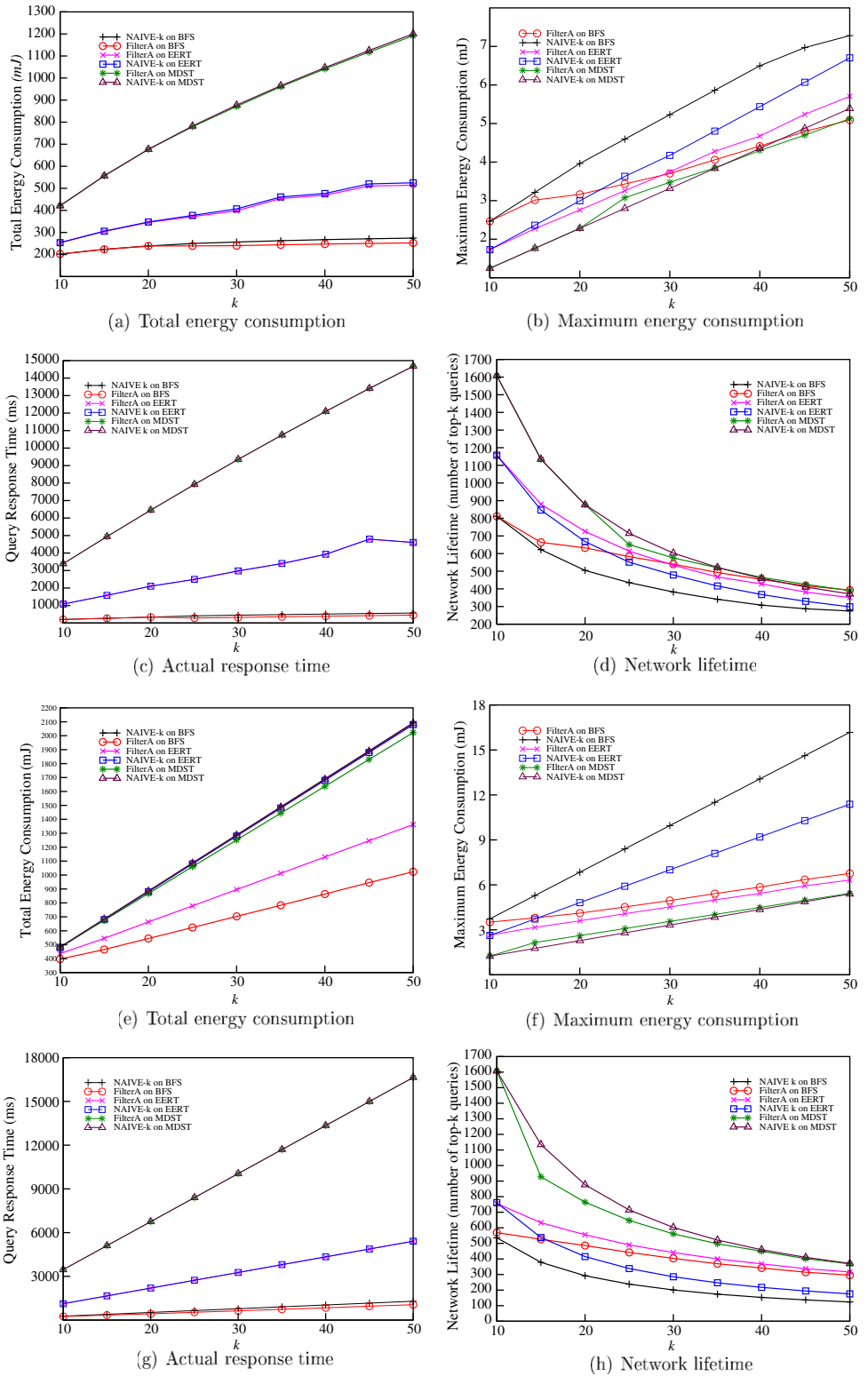

Fig. 1. The performance of different routing algorithms and various routing trees for top-k query evaluation in terms of the total energy consumption, the maximum energy consumption among the nodes, the query response time, and the network lifetime, assuming that the network contains 500 sensor nodes and each node contain either one reading (figures (a)–(d)) or k readings (figures (e)–(h)) with $10 \leqslant k \leqslant 50$.

We set the initial energy capacity at each sensor node to be 2000 mJ. For the top-k query problem, we assume that each sensor node contains one or k readings initially, $10 \leqslant k \leqslant 50$. In other words, the problem is to identify up to the top 10% sensor nodes with the highest readings among the 500 sensor nodes in G. The sensing readings are simulated, using a real trace data, provided by the Live from Earth to Mars (LEM) project at the University of Washington [12]. We use the temperature traces logged by the station at the University of Washington covering a period from January 1, 2005 to December 31, 2005. From this large volume of TEMP trace data where there is a reading per minute and each reading is represented by $l = 4$ bytes. Suppose that $l_H = 8$ bytes. We extracted 1000 subtraces by partitioning the original trace into 1000 consecutive segmentations and each segmentation contains 500 consecutive readings within an interval of 500 min. Thus, each subtrace contains 500 readings that are assigned to the 500 sensor nodes in the network randomly. In all our experiments, we define the network lifetime as the number of top-k queries answered by the network until the network is unable to answer any further top-k query due to the failure of one of its nodes. Each value in all charts is the mean of 1000 results obtained by running an algorithm 1000 times and each time the input of the algorithm is a different segment of subtrace readings.

6.2. Impact of different size k on the network lifetime

We first evaluate the impact of different algorithms for top-k query evaluation on the network lifetime by varying the size of k. We evaluate the performance of algorithms `NAIVE`-k and `FilterA` on three different routing trees: the TAG routing tree [13], which actually is a Breadth-First-Search (BFS) tree; MDST; and the tree obtained by applying algorithm `Find_Routing_Tree` on the initial tree MDST until there is no further improvement in the resulting tree, we refer this resulting tree as an energy-efficient routing tree (EERT for short). The performance of various algorithms is depicted by Fig. 1(a)–(h).

We first consider the case where each node contains just one reading initially. Fig. 1(a) implies that the total energy consumption among the nodes is proportional to the size of k. With the increase of k, the total energy consumption by using MDST is significantly higher than that by using EERT and BFS, and the total energy consumption by BFS is the minimum one among the trees. It can also be seen from Fig. 1(b) that the maximum energy consumption by applying algorithm `FilterA` on BFS is less than that by applying the same algorithm on EERT when $k \geqslant 30$, which is explained as follows.

Let Δ_{EERT} be the maximum degree of nodes with no less than k descendants in EERT and Δ_{BFS} the maximum degree of BFS. When k is small in comparison with the network size n, for the top-k query using the proposed filtering algorithm on either of the two trees, the maximum energy consumption among the nodes happens at the nodes with the maximum degree in the corresponding tree that each child of the nodes has at least k descendants. When the value of k becomes larger without changing the network size n, the average number k' of descendants of each child of the maximum degree node will be less than k, i.e., $k' < k$. Meanwhile, it is known that the reception energy consumption at a node is proportional to the number of children and the average number of top-k readings of the children, assuming that the number of descendants of some children is no more than k. Thus, the maximum energy consumption among the nodes in BFS or EERT for top-k query evaluation is proportional to $\Delta_{BFS}k_{BFS}$ or $\Delta_{EERT}k_{EERT}$, where k_{BFS} and k_{EERT} are the average numbers of top-k readings of children at a node with the maximum degree in trees BFS and EERT respectively. Clearly, $k_{EERT} > k_{BFS}$ and $1 \leqslant k_{EERT}, k_{BFS} < k$, because $\Delta_{EERT} < \Delta_{BFS}$ and the network size is given. The maximum energy consumption derived by EERT is larger than that by BFS if Δ_{EERT}-$k_{EERT} > \Delta_{BFS}k_{BFS}$. In other words, the claim holds if

$$\frac{\Delta_{EERT}}{\Delta_{BFS}} > \frac{k_{BFS}}{k_{EERT}}. \tag{4}$$

By Inequality (4), once $\frac{\Delta_{EERT}}{\Delta_{BFS}}$ is fixed, it can be seen that the increasing rate of k_{EERT} is faster than that of k_{BFS} with the growth of k. In other words, let k become $k + \delta k$, correspondingly, k_{BFS} and k_{EERT} become $k_{BFS} + \delta k_{BFS}$ and $k_{EERT} + \delta k_{EERT}$, and $\delta k_{EERT} > \delta k_{BFS}$. Thus, $\frac{k_{BFS}}{k_{EERT}} > \frac{k_{BFS}+\delta k_{BFS}}{k_{EERT}+\delta k_{EERT}}$, and Inequality (4) continues to hold, with the growth of k. Note that in our experiment, $\Delta_{BFS} = 11$ while $\Delta_{EERT} = 5$, and $n = 500$ nodes are deployed. The maximum degree of nodes in BFS contains no more than $5 * k = 5*30 = 150$ descendants when $k \geqslant 30$. With maintaining the same response time, Fig. 1(f) implies that algorithm `FilterA` significantly outperforms algorithm `NAIVE`-k in terms of the maximum energy consumption among the nodes, no matter which routing tree is used. This means a much longer network lifetime will be derived by algorithm `FilterA` from Fig. 1(c). Meanwhile, the difference of network lifetime between the two algorithms is insignificant when apply them on MDST, because in our experiments, the maximum degree of nodes in MDST is two, for which algorithm `FilterA` does not have much effect on the removal of some readings from transmission. Fig. 1(c) and (g) show that the query response time grows, with the increase of the value of k. Although applying algorithm `NAIVE`-k or algorithm `FilterA` on MDST leads to a reasonably longer network lifetime (see Fig. 1(d) and (h)), the corresponding query response time is the longest as well. Furthermore, the growth rate of the response time by MDST is the fastest one. On the other hand, although applying algorithm `NAIVE`-k on BFS leads to the fastest query response time, it will also result in the shortest network lifetime (see Figs. 1(d) and (h)). Obviously, both BFS and MDST represent two extreme cases in terms of the tradeoff between the query response time and the network lifetime. To obtain a reasonably longer network lifetime and meet a specified response time constraint Γ simultaneously, applying algorithm `FilterA` on EERT is a nice approach. Through this experiment, we demonstrate that the query response time and the network lifetime are highly correlated, and the query response time is proportional to the network lifetime. That is, the shorter the response time, the shorter the network lifetime. Figs. 1(e)–(h) plot the performance of various algorithms

on different routing trees with an assumption that each node contains k instead of one reading, which is similar to the one where each node contains one reading, omitted.

6.3. Impact of the maximum node degree of different routing trees on the network lifetime

We then investigate the impact of the maximum degree of nodes Δ in different routing trees on the network lifetime while meeting the given query response time constraint, using evaluation algorithm NAIVE-k or FilterA. Assume that these routing trees meet the response time constraint and $k = 50$. The performance curves are plotted in Figs. 2(a)–(h), where "$D\ \Delta$" at the x-coordinate represents a routing tree with maximum degree of Δ, $3 \leqslant \Delta \leqslant 7$, which is constructed as follows.

We start from MDST initially. Let Δ be the maximum node degree of the current routing tree, we set the maximum degree of the next routing tree to be $\Delta + 1$, the next routing tree can be obtained by running the proposed algorithm for finding routing trees without iterations. We continue this procedure until there is no further improvement to the current routing tree. Let EERT be the final tree. Thus, a series of routing trees is obtained with different maximum degrees of nodes.

We assume that each node contains k readings initially. Fig. 2(a) illustrates that in terms of the total energy consumption, there is no difference among different routing trees on which algorithm NAIVE-k is applied, while Fig. 2(b) indicates that the maximum energy consumption derived by different routing trees is proportional to the maximum degree of nodes in corresponding routing trees when algorithm NAIVE-k is applied. In contrast, there are no significant differences among different routing trees on which algorithm FilterA is applied. Furthermore, algorithm FilterA significantly outperforms algorithm NAIVE-k in terms of the maximum energy consumption among the nodes (see Fig. 2(b)), which means that it leads to a significantly longer network lifetime (see Fig. 2(d)) when they both apply on the same routing tree. Fig. 2(c) implies that the query response time is inversely proportional to the maximum degree of a routing tree, regardless of which evaluation algorithm used. It is noticed from this figure that the query response time is no longer reduced when the maximum degree of nodes in a routing tree reaches a certain threshold. In our case, the maximum degree of nodes in EERT is no less than 5, where the maximum degree of nodes in the network of 500 nodes is 24 and the maximum degree of BFS is 11. Fig. 2(c) and (d) demonstrate that the query response time is proportional to the network lifetime, no matter which algorithm is applied. A longer response time implies a longer network lifetime. On the other hand, given the same response time constraint and the same routing tree, the network lifetime derived by algorithm FilterA is significantly longer than that by algorithm NAIVE-k.

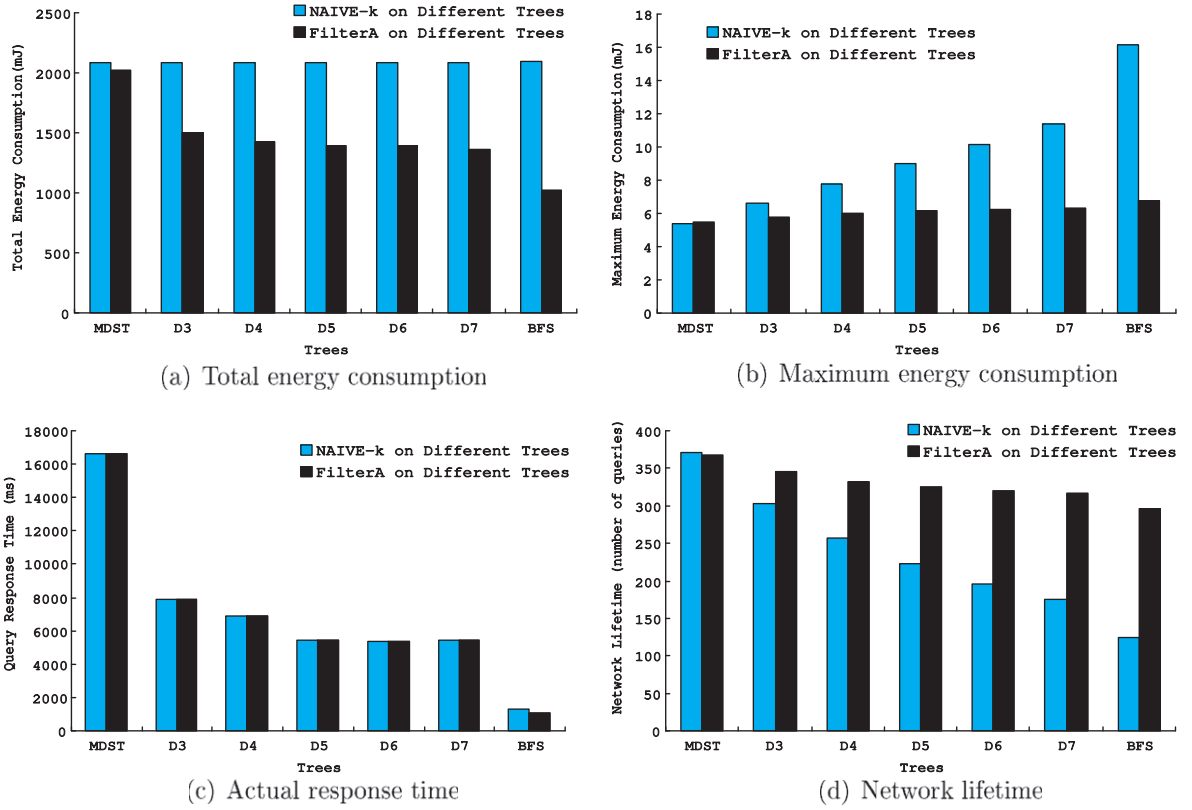

Fig. 2. The performance of different routing trees for top-k query evaluation in terms of the total energy consumption, the maximum energy consumption among the nodes, the query response time, and the network lifetime, assuming that the network contains 500 nodes and each node contains $k = 50$ readings.

7. Conclusions

We have studied the top-k query problem in sensor networks subject to specified response time constraints. We first formulated the problem, followed by introducing a cost model of energy consumption of top-k query evaluation and the query response time definition. We then proposed a novel joint optimization framework, consisting of finding a routing tree in the network and devising a filter-based evaluation algorithm for top-k query evaluation on the found tree. We finally conducted extensive experiments by simulations on a real dataset to evaluate the performance of the proposed algorithms, in terms of the total energy consumption, the maximum energy consumption among the nodes, the query response time, and the network lifetime. The experimental results showed that there is a non-trivial tradeoff between the query response time and the network lifetime. Given the query response time constraint, the proposed joint optimization framework can prolong the network lifetime significantly.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Communications Magazine (2002) 102–114.
[2] J. Anderson, L. Hong, Sensor resource management driven by threat projection and priorities, Information Sciences 178 (2008) 2007–2021.
[3] B. Babcock, C. Olston, Distributed top-k monitoring, in: Proceedings of ACM SIGMOD, ACM, 2003, pp. 28–39.
[4] P. Cao, Z. Wang, Efficient top-k query calculation in distributed networks, in: Proceedings of ACM PODC, ACM, 2004, pp. 206–215.
[5] J.-H. Chang, L. Tassiulas, Energy conserving routing in wireless ad hoc networks, in: Proceedings INFOCOM'00, IEEE, 2000, pp. 22–31.
[6] Crossbow Inc., MPR-Mote Processor Radio Board Users Manual.
[7] R. Fagin, A. Lotem, N. Naor, Optimal aggregation algorithms for middleware, in: Proceedings of ACM PODS, ACM, 2001, pp. 102–123.
[8] M. Fürer, B. Raghavachar, Approximating the minimum degree spanning tree to within one from the optimal degree, in: Proceedings 3rd ACM-SIAM Symp. on Discrete Algorithms, ACM, 1992, pp. 317–324.
[9] M.R. Garey, D.S. Johnson, Computer and Intractability: A Guide to the Theory of NP-completeness, W.H. Freeman, San Francisco, 1979.
[10] Z. He, B.S. Lee, X.S. Wang, Aggregation in sensor networks with a user-provided quality of service goal, Information Sciences 178 (2008) 2128–2149.
[11] S. Lindsey, C.S. Raghavendra, K.M. Sivalingam, Data gathering algorithms in sensor networks using energy metrics, IEEE Transactions on Parallel and Distributed Systems 13 (2002) 924–935.
[12] Live from Earth and Mars (LEM) Project, 2006. <http://www-k12.atoms.washington.edu/k12/grayskies>.
[13] 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 36 (2002) 131–146.
[14] S. Madden, M.J. Franklin, J.M. Hellerstein, W. Hong, The design of an acquisitional query processor for sensor networks, in: Proceedings SIGMOD'03, ACM, 2003, pp. 491–502.
[15] S. Michael, P. Triantafillou, G. Weikum, A framework for distributed top-k query algorithms, in: Proceedings of 31th Conference on VLDB, 2005, pp. 637–648.
[16] S. Madden, R. Szewczyk, M.J. Franklin, D. Culler, Supporting aggregate queries over ad hoc wireless sensor networks, in: Proceedings 4th IEEE Workshop on Mobile Computing and System Applications, IEEE, 2002, pp. 49–58.
[17] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, J. Anderson, Wireless sensor networks for habitat monitoring, in: Proceedings of Int'l Workshop on Wireless Sensor Networks and Applications, ACM, 2002, pp. 88–97.
[18] The Network Simulator-ns2, 2006. <http://www.isi.edu/nsnam/ns>.
[19] G. Pottie, W. Kaiser, Wireless Sensor Networks, Commun. ACM, vol. 43, ACM, 2000. pp. 51–58.
[20] B. Prabhakar, E. Uysal-Biyikoglu, A.E. Gamal, Energy-efficient transmission over a wireless link via lazy packet scheduling, in: Proc of INFOCOM'01, IEEE, 2001, pp. 386–394.
[21] Q. Ren, Q. Liang, Energy and quality aware query processing in wireless sensor database systems, Information Sciences 177 (10) (2007) 2188–2205.
[22] C. Schurgers, O. Aberhorne, M.B. Srivastava, Modulation scaling for energy-aware wireless microsensor networks, in: Proceedings of ISLPED, IEEE, 2001.
[23] A. Silberstein R Braynard, C. Ellis, K. Munagala, J. Yang, A sampling-based approach to optimizing top-k queries in sensor networks, in: Proceedings of 22nd Int'l Conf. on Data Engineering, IEEE, 2006.
[24] M. Wu, J. Xu, X. Tang, W-C. Lee, Monitoring top-k query in wireless sensor networks, in: Proceedings of 22nd Int'l Conf. on Data Engineering, IEEE, 2006.
[25] M. Wu, J. Xu, X. Tang, W-C. Lee, Top-k monitoring in wireless sensor networks, IEEE Transactions on Knowledge and Data Engineering 19 (7) (2007) 962–976.
[26] Y. Yao, J. Gehrke, The cougar approach to in-network query processing in sensor networks, ACM SIGMOD Record 31 (2002) 9–18.
[27] Y. Yao, J. Gehrke, Query processing for sensor networks, in: Proc of the 2003 Conf. on Innovative Data Systems Research, 2003.
[28] Y. Yu, B. Krishnamachari, V.K. Prasanna, Energy-latency tradeoffs for data gathering in wireless sensor networks, in: Proceedings of INFOCOM, IEEE, 2004, pp. 244–255.
[29] D. Zeinalipour-Yazti, Z. Vagena, D. Gunopulos, V. Kalogeraki, V. Tsotras, M. Vlachos, N. Koudas, D. Srivastava, The threshold join algorithm for top-k queries in distributed sensor networks, in: Proceedings of DMSN in VLDB Conference, 2005, pp. 61–66.