Progressive Skyline Query Processing in Wireless Sensor Networks

Baichen Chen  Weifa Liang
Department of Computer Science
Australian National University
Canberra, ACT 0200, Australia

Abstract—With the further development of sensor techniques in wireless sensor networks (WSNs), it is becoming urgent that they should be able to support complicated queries like skyline query for multi-preference and decision making. In this paper, we consider skyline query evaluation in WSNs by devising evaluation algorithms for finding skyline points on a dataset progressively. The core techniques adopted are to partition the dataset into several disjoint subsets and output the skyline points by examining each subsequent subset progressively, using some of the skyline points obtained so far to filter out those unlikely skyline points in the current processing subset. We finally conduct extensive experiments by simulations to evaluate the performance of the proposed algorithms on synthetic and real datasets. The experimental results show that the proposed algorithms outperform existing algorithms significantly in network lifetime prolongation.

Keywords: wireless sensor network, progressive algorithms, skyline query, query optimization, energy conservation

I. INTRODUCTION

To support data query processing in wireless sensor networks (WSNs), several DB systems like TinyDB [13] and DB Cougar [25] have been developed in the past years. These DB systems enable supporting some basic operators including $\text{SUM}$, $\text{MIN}$, $\text{AVG}$, etc, due to the miniature hardware constraints imposed on sensors such as limited storages, powered by energy-limited batteries, slow processing capabilities, small communication bandwidths. With the further development of hardware techniques in sensors and WSNs applications, it is becoming urgent that WSNs are also able to support more complicated queries like self-join [26], top-$k$ [19] and skylines [24]. In this paper we focus on skyline query processing in WSNs, which is a popular one in modern databases for multi-criteria decision making that has been received much attention recently by the database community. It can be formally defined as follows. For two $d$-dimensional data points $p$ and $q$, point $p$ is dominated by point $q$, denoted by $q \prec p$, if $q$ is no worse than $p$ on all $d$ dimensions and $q$ is strictly better than $p$ on at least one dimension. Given a set $S$ of $d$ dimensional data points, a point $p$ in $S$ is a skyline point of $S$ if $p$ is not dominated by any other points in $S$. The skyline query on $S$ retrieves all the skyline points in $S$.

Skyline queries in WSNs can be used to monitor the extreme sensing data under multiple criteria. For example, scientists can deploy a WSN to monitor air pollution of a region of interest, where the sensors sense the concentration of poisonous gases like $\text{CO}$ or $\text{SO}_2$. The places with high concentration of either $\text{CO}$ or $\text{SO}_2$ are regarded to suffer the serious air pollution. A skyline query on the WSN can identify such places for environmental monitoring and improvement purpose. Another example is using WSNs to monitor bushfires, where a sensor in the WSN can sense temperature, humidity and smoke density about its vicinity. In bushfire, the fieriest fire will cause a place with extreme high temperature, low humidity and high smoke density. The places with low temperatures but high smoke densities are also dangerous. To extinguish bushfire, fire fighters issue a skyline query to identify such places.

A. Related Work

Most previous studies on skyline query focused on the centralized databases by assuming that the data is stored in a centralized database [2], [3], [8], [9], [16], [20]. The other work dealt with various skyline queries under other computational environments, including skyline processing over data streaming [10], top-$k$ skyline with the maximum number of dominated points [11], spatial skyline [12], skyline with partially ordered domains [5], and probabilistic skyline on a set of uncertain data points [17]. Beyond the studies on centralized databases, skyline query has also been exploited in decentralized databases such as the World Wide-Web [1], CAN P2P network [21], BATON P2P network [6] and P2P systems with different topological structures [22].

Although extensive studies on skyline query in traditional databases have been conducted in recent years, these existing algorithms are not applicable to WSNs due to the following unique constraints imposed on WSNs. First, the centralized data structures like $R$-tree employed in centralized databases for skyline query processing no longer exist in WSN environments so that the algorithms based on these centralized data structures are inapplicable to WSNs. Second, sensors have limited storages and processing capabilities in comparison with powerful computers, there is no such a powerful centralized sensor in WSNs that is able to communicate with the other sensors as that in a traditional distributed system. Finally, unlike known algorithms in centralized and distributed databases focusing on optimizing query response time, space and the number of packets, the optimization objective of skyline processing in sensor networks is the energy consumption of answering the query, while wireless communication is the dominant part of all types of energy consumptions of sensors [13], [18].
Skyline query evaluation on WSNs has also been exploited [7], [23], [24] recently. For example, Huang et al. [7] dealt with a constrained skyline query problem on MANETs by devising a single point filter-based evaluation algorithm that is easily extended to WSNs. Xin et al [24] studied the problem by devising two filter-based algorithms. One is the single point filter-based algorithm TF and another is the grid filter-based algorithm GI. Algorithm TF chooses the point that dominates the maximum number of points as the filter, assuming that the data distribution density is given beforehand, while algorithm GI exploits the grid partition of data space and generates a grid filter. Liang et al. [23] recently proposed a new filter-based algorithm which consists of multiple rather than single point as the filter, in which each sensor sends part of its local skyline points by a greedy algorithm to its parent and the root broadcasts the received points as the global certificate obtained through in-network aggregation. The points that cannot pass through the certificate will be filtered out from transmission. However, the existing algorithms for skyline queries in WSNs have their own limitations. For example, the filter in [7] is only determined by the local rather than global information, which leads to inefficiency of the filter. The authors in [24] assume that the density function of data is known beforehand, which may be too restrictive in the real world. Moreover, existing algorithms for skyline query in WSNs mainly focus on the optimization of the total energy consumption by ignoring the maximum energy consumption among the sensors. However, the maximum energy consumption among the sensors is another important optimization parameter which plays the key role in determining the network lifetime [14]. Notice that in sensor networks, the sensors near to the base station exhaust their batteries first, which renders the rest of sensors disconnected from the base station. Consequently, the base station cannot receive the data from the rest of sensors. Therefore, a desired algorithm for skyline query evaluation should not only optimize the total energy consumption but also the maximum energy consumption among the sensors. Above all, the design of energy-efficient algorithms for skyline query in WSNs poses great challenges.

B. Contributions

Our major contributions in this paper are as follows. Two algorithms for evaluating skyline query are devised, which find the skyline points progressively. The novel techniques behind are to partition the dataset into disjoint subsets, followed by returning the skyline points through examining each subsequent subset progressively, using some found skyline points so far as a filter to filter out those unlikely skyline points in the currently processing subset from transmission. We finally conduct extensive experiments by simulations on both synthetic and real datasets. The experimental results show that the proposed algorithms significantly outperform existing algorithms in terms of various performance metrics.

The remainder of the paper is organized as follows. Section 2 introduces the wireless sensor network model and problem definition, followed by giving an important observation that is the cornerstone of the proposed algorithms. Sections 3 and 4 propose two algorithms for progressively evaluating skyline queries. Extensive experiments are conducted to evaluate the performance of the proposed algorithms in Section 5, and the conclusions are given in Section 6.

II. Preliminaries

A. System Model

We consider a wireless sensor network consisting of \( n \) stationary sensors randomly deployed in a region of interest, and each sensor measures \( d \) attribute values. There is a base station with unlimited energy supply serving as the gateway between the sensor network and the users. All sensors can communicate with the base station via one or multi-hop relay, and a sensor can communicate with the sensors located within its transmission range. We further assume that the transmission range of each sensor is identical. The battery-powered sensors can not only sense and collect data from their vicinities but also process and transmit the data to their neighbors. To transmit a message containing \( k \) bytes of data from one sensor to another, the amounts of transmission energy consumed at the sender are \( p_t R k \), and the amounts of reception energy consumed at the receiver are \( p_r r_c k \), where \( p_t \) and \( p_r \) are the sum of energy overhead on handshaking and transmitting and receiving header part of the message, \( R \) and \( r_c \) are the amounts of transmission and reception energy per byte [23]. Every \( d \)-dimensional data point is represented by \( 4d \) bytes in this paper. In our cost model, the computational energy consumption at each sensor is ignored, because the energy consumption of local computation is at least several orders of magnitude less than the energy consumption on radio communication, which can be witnessed by [13], one bit of data transmitted by the radio takes as much energy as executing 1000 CPU instructions. Therefore, unless otherwise specified, we only consider the communication energy consumption in the performance evaluation section.

B. Problem Definition

Given a wireless sensor network \( G(V,E) \) with base station \( r \), where \( V \) is the set of sensors and \( E \) is the set of links. Assume that each sensor \( v \) in \( V \) contains a set of \( d \)-dimensional points \( P(v) \) generated during a given time interval, and \( P = \bigcup_{v \in V} P(v) \) forms the entire dataset. The skyline query on the snapshot set \( P \) is to find a subset of \( P, SK(P) \), in which the points cannot be dominated by any other points in \( P \). Without loss of generality, we assume that the value range of each dimension of a point is within \( [0, +\infty) \) and the whole \( d \)-dimensional data space \( DS = \{[0, +\infty), [0, +\infty), ..., [0, +\infty)\} \) is the union of subspaces distributed at the \( |V| \) sensors. If a data space contains the points with negative values, it is easy to transform the points in this space to another data space \( DS \) which contains the points with non-negative values through the coordinate transformation, and then transform the query results under the data space \( DS \) back to the query results under the original data space. In different contexts, the term “better” can be interpreted as either the “smaller” or the “larger”. Without loss of generality, we say that “the better” means “the smaller”.

\( V \)
\( E \)
\( P \)
\( DS \)
\( SK(P) \)
Energy-efficient skyline query evaluation in sensor networks can be implemented through in-network processing paradigm [14], [25], that is a routing tree \( T \) rooted at the base station \( r \) and spanning all sensors is employed for skyline query evaluation. The query evaluation on \( T \) consists of distribution stage, to push the query down to each sensor along the paths of tree; and collection stage, to collect the sensed data from children to parent and eventually to the base station through multiple-hop relay. Unless otherwise specified, in the rest of this paper, we assume that such a tree exists and the tree will be used for query evaluation.

A well-known algorithm for skyline query evaluation is algorithm Skyline Merge which proceeds as follows. If sensor \( r \) is a leaf sensor, it sends the skyline of local points to its parent; otherwise, sensor \( v \) computes the skyline on the set of the points at \( v \) which includes the points generated at \( v \) and the points forwarded from its children, and \( v \) then transmits the new skyline to its parent. Finally, the base station \( r \) calculates the skyline of the received points that is the skyline on the set of all the points in \( P \).

The following important observation is the cornerstone of all proposed algorithms in the rest of the paper.

**Definition 1:** [8], [16] Suppose \( p = (p_1, p_2, ..., p_d) \) is a \( d \)-dimensional point, the radius of \( p \) \( R(p) \) is defined as the Euclidean distance between \( p \) and the origin \( o \), i.e., \( R(p) = \sqrt{\sum_{i=1}^{d} p_i^2} \).

**Observation 1:** [16] Let \( R(p) \) and \( R(q) \) be the radii of points \( p \) and \( q \). If \( R(p) \leq R(q) \), point \( p \) cannot be dominated by point \( q \).

### III. FIXED PARTITION ALGORITHM

In this section, an evaluation algorithm is proposed, which partitions the dataset into \( k \) disjoint subsets and proceeds in a number of iterations \( k \). The value of \( k \) is given beforehand. In each iteration, the proposed algorithm returns the skyline points progressively and chooses some of the found skyline points so far for the global filter to filter out the unlikely skyline points in the rest of non-examined subsets from transmission. In the following we describe how to partition the dataset through partition radius.

**A. Fixed Dataset Partition**

Given a \( d \)-dimensional dataset \( P \) consisting of all the points in the network, denote by \( R(P)_{\text{max}} = \max\{R(p) \mid p \in P\} \) and \( R(P)_{\text{min}} = \min\{R(p) \mid p \in P\} \) the maximum and minimum radii of dataset \( P \). The basic idea behind the proposed algorithm is to partition the dataset \( P \) into \( k \) disjoint subsets, \( P_1, P_2, ..., P_k \), such that \( R(P_i)_{\text{max}} < R(P_{i+1})_{\text{min}} \) for all \( 1 \leq i < k \), where \( k \geq 1 \) is a given integer. In the \( i \)-th iteration, the proposed algorithm examines the points in \( P_i \) and finds the new skyline \( SK_i \) in which each point is not dominated by the found skyline points so far. The skyline on set \( P \) is the union of the set of newly found skyline in each iteration, i.e., \( SK(P) = \bigcup_{i=1}^{k} SK_i \).

The algorithm proceeds to obtain \( R(P)_{\text{min}} \) and \( R(P)_{\text{max}} \) on dataset \( P \), using in-network aggregation. Having \( R(P)_{\text{min}} \) and \( R(P)_{\text{max}} \), it first generates a series of \( k \) ascending radii \( R(P_i)_{\text{max}} \) for all \( 1 \leq i \leq k \), which is an arithmetic or geometric sequence. To generate an arithmetic sequence, suppose \( R(P)_{\text{max}} = R(P)_{\text{min}} + a + 2a + ... + ka \) and the approximate value of \( a = \frac{2R(P)_{\text{max}} - R(P)_{\text{min}}}{k+1} \). Thus, \( R(P_i)_{\text{max}} = R(P)_{\text{min}} + i \times a \) and \( R(P_i)_{\text{min}} = R(P)_{\text{min}} + i \times a \).

To generate a geometric sequence, suppose \( R(P)_{\text{max}} = R(P)_{\text{min}} + q + q^2 + ... + q^k \). Thus, \( R(P_i)_{\text{max}} = R(P)_{\text{min}} + q^i \) and \( R(P_i)_{\text{min}} = R(P)_{\text{min}} + q^i \). According to the arithmetic or geometric series of \( R(P_i)_{\text{max}} \), the dataset \( P \) can be partitioned into \( k \) disjoint subsets \( P_1, P_2, ..., P_k \) and the radii of the points in set \( P_i \) are within \( [R(P_{i-1})_{\text{max}}, R(P_i)_{\text{max}}) \), where \( R(P_i)_{\text{max}} \) is the partition radius of subset \( P_i \) of \( P \), \( 1 \leq i \leq k \).

It then proceeds with \( k \) iterations. Denote by \( SK(S) \) the skyline on set \( S \). Denote by \( LSK(v) \), the skyline of the points at sensor \( v \) and the points received from the children of \( v \) and \( LF(v) \), the local filter of sensor \( v \) in the \( i \)-th iteration that consists of several points. Initially, \( LF(v) = 0 \) and \( LSK(v) = SK(P(v)) \) if \( v \) is a leaf sensor, where \( P(v) \) is the set of points generated at sensor \( v \). We refer to the algorithms that partition the dataset using the arithmetic or geometric series as algorithm \( a\text{-FDP} \) or \( g\text{-FDP} \) (\( F_{\text{Fixed Dataset Partition}} \)), respectively. In the \( i \)-th iteration each of the algorithms proceeds as follows.

Every sensor \( v \) first filters out the points at its own that are dominated by the local filter \( LF(v) \). If \( v \) is a leaf sensor, it sends the points in \( LSK(v) \), whose radii are no greater than \( R(P_i)_{\text{max}} \) to its parent. Otherwise, sensor \( v \) calculates \( LSK(v) \), of the points at sensor \( v \) and the points received from its children and transmits the points in \( LSK(v) \) whose radii are no greater than \( R(P_i)_{\text{max}} \) to its parent. In the end, the base station \( r \) calculates the skyline \( LSK(r) \) on all the received points. Recall that \( SK_i = \{p \mid p \in LSK(r), q \neq p, q \in \bigcup_{j=1}^{i} SK_j\} \) as the newly found skyline points in the \( i \)-th iteration. \( SK(\bigcup_{j=1}^{i} P_j) = \bigcup_{j=1}^{i} SK_j \cup SK_i \). Finally, some points in \( SK_i \) are chosen and broadcast to the sensors in the sensor network for filtering purpose. Denote by \( GS_F \), the set of skyline points broadcast in the \( i \)-th iteration. Every sensor then updates its local filter with \( GS_F \), i.e., \( LF(v)_{i+1} = LF(v)_{i} \cup GS_F \). Having performed \( k \) iterations, the skyline on dataset \( P \) is \( \bigcup_{i=1}^{k} SK_i \).

In the following we detail which points in \( SK_i \) to be chosen in each iteration for algorithm \( a\text{-FDP} \) or \( g\text{-FDP} \).

**B. Choosing Skyline Points for Global Filter**

Broadcasting some of the newly found skyline points as a global filter aims to filter out those unlikely skyline points in the rest of unprocessed subsets from transmission in future iterations. In the real world, it is impossible to figure out the exact number of points filtered out by the chosen skyline points before these chosen skyline points are broadcast, because there is not any knowledge of data distribution at sensors. Instead, the volume of dominance region of a point can be used to represent its filtering “gain” - the number of points is dominated by the point. The dominance region of a point \( p \) is the region in which any point is dominated by \( p \). Having obtained \( SK_i \) at the base station \( r \), a simple way to update
its local filter of a sensor \( v \) is to broadcast all newly found global skyline points to each sensor. However, this naive approach will incur much more energy overhead than needed, due to the fact that the dominance regions of most found skyline points are overlapping with each other, and only a few of them dominate most of the whole dominance region. On the other hand, if the newly found global skyline points are not broadcast, the local filter of each sensor will become inefficient due to lack of the updated global information. Consequently, the local filter of each sensor cannot filter out as many unlikely skyline points as possible, and the sensor will incur excessive energy overhead on unlikely skyline points transmission. We thus propose a method to tradeoff the energy consumption between the filter points broadcasting and the unlikely skyline points transmission without filtering, based on the volume of the efficient dominance region of each point. The efficient dominance region of a point \( p \) is the subspace of the dominance region of \( p \), which is not covered by the dominance regions of the previous found skyline points and all the points inside are not examined yet.

The intuition of choosing which skyline points in \( SK_i \) is that efficient dominance regions of skyline points obtained in the current iteration are in the margin space of the dominance regions of found skyline points in previous iterations, which is illustrated by Fig. 1. Suppose that there are two skyline points \( p_0 \) and \( p_1 \) obtained in the first iteration and \( Region D \) is the union of dominance regions of \( p_0 \) and \( p_1 \). Points \( p_2 \) and \( p_3 \) are the found skyline points in \( SK_2 \) and \( r_2 = R(P_{2\text{max}}) \). Following algorithm a-FDP or g-FDP, after performing the second iteration, the dominance region within the fan region \( S(Orzr^2) \) will be useless for filtering purpose, since all the points in the region have already been examined. The efficient dominance regions of \( p_2 \) (Region ADGH) and \( p_3 \) (Region BCEFH) are actually located in the margin space of region \( D \) (between region \( D \) and \( X \) axis), because most of the dominance regions of \( p_2 \) and \( p_1 \) have been covered by \( Region D \) and \( Region S \). From Fig. 1 we can observe that Region BCEFH of \( p_3 \) covers most of Region ADGH of \( p_2 \) except Region ABC, which implies that only broadcasting \( p_2 \) will lead to the same filtering gain as broadcasting both \( p_2 \) and \( p_3 \). In higher dimensional datasets, it is more complicated to calculate the volume of the efficient dominance regions of points. We extend our analysis to a more general case by developing a greedy approach, which is detailed as follows.

Denote by \( EDR(p_j) \) an approximate volume of the efficient dominance region of point \( p \) at the \( j \)th dimension. Let \( GSF_i \) be the set of chosen skyline points to be broadcast after the \( i \)th iteration. Given a set \( P \) of \( d \) dimensional points, the maximum point \( MAX = (\text{max}_1, \ldots, \text{max}_d) \) is a virtual point, where \( \text{max}_j \) is the maximum value at the \( j \)th dimension of the points in \( P \) and the minimum skyline point \( MinSK(i) = (\text{min}_1(SK_i), \ldots, \text{min}(SK_i)) \) is another virtual point, where \( \text{min}_j(SK_i) \) is the minimum value at the \( j \)th dimension of all found skyline points in the first \( i \) iterations. Let \( \text{margin}(p_i) = \text{min}_j(SK(i-1))_{p_j} \) represents how far point \( p \) from the dominance regions of found skyline points in previous iterations at the \( j \)th dimension. The approximate volume of the efficient dominance region of point \( p = (p_1, p_2, \ldots, p_d) \) in \( SK_i \) at the \( j \)th dimension is \( EDR(p_j) = \text{margin}(p_j) \times (\prod_{k=1, k \neq j}^{d} (\text{max}_k - p_k) - \prod_{k=1, k \neq j}^{d} (R(P_{1\text{max}}) - p_k)) \), where \( R(P_{1\text{max}}) \) is the partition radius of subset \( P \) of \( P \). For each dimension \( j \), the point \( p \) in \( SK_i \) with the maximum value of \( EDR(p_j) \) is chosen and added to \( GSF_i \), if \( p \not\in GSF_i \) and \( EDR(p_j) > 0 \). Finally, at most \( d \) skyline points (at most one point chosen in each dimension) are chosen and broadcast into the sensor network for local filter updating after the \( i \)th iteration. The virtual point \( MAX \) can be obtained using in-network processing within the first iteration, and the points with the minimum value at each dimension are added to \( GSF_i \). The following is the pseudocode of determining the global filter in the \( i \)th iteration.

Algorithm 1 GSF \((i, SK_i)\)

begin
\[ GSF_i = \emptyset; \]
\[ \text{if } i = 1 \text{ then} \]
\[ \text{foreach dimension } j \left(1 \leq j \leq d\right) \text{ do} \]
\[ \text{if } (p_j = \text{min}_j(q_j, q \in SK_i)) \text{ and } (p \not\in GSF_i) \text{ then} \]
\[ GSF_i \leftarrow GSF_i \cup \{p\}; \]
\[ \text{end} \]
\[ \text{else} \]
\[ \text{foreach dimension } j \left(1 \leq j \leq d\right) \text{ do} \]
\[ \text{if } (EDR(p_j) = \text{max}(EDR(q_j), q \in SK_i)) \text{ and } (EDR(p_j) > 0) \text{ and } (p \not\in GSF_i) \text{ then} \]
\[ GSF_i \leftarrow GSF_i \cup \{p\}; \]
\[ \text{end} \]
\[ \text{end} \]
\[ \text{update the virtual point } MinSK(i) \text{ and broadcast } GSF_i; \]
end

C. Correctness of Fixed Partition Algorithm

The rest is to show the correctness of algorithms a-FDP and g-FDP, i.e., we show that \( SK(P) \) obtained after \( k \) iterations is the skyline on set \( P \) by the following theorem.

Theorem I: For a dataset \( P \), the skyline \( P, SK(P) = \bigcup_{i=1}^{k} SK_i \), where \( SK_i \) is the new skyline points delivered by algorithm a-FDP or g-FDP in the \( i \)th iteration, \( 1 \leq i \leq k \).

Proof: Assume that there is a point \( p \) with \( R(p) \in (R(P_{i-1\text{max}}), R(P_{i\text{max}})) \). We prove the claim through proving that if \( p \not\in SK_i \), \( p \) must be dominated by other points; otherwise, \( p \) cannot be dominated by any other points in \( P \).
Clearly, only the points whose radii are in the range between $R(P_{i-1})_{\text{max}}$ and $R(P_i)_{\text{max}}$ are possible to be added to $SK_i$, which implies that the subset $P_i$ of $P$ contains the points with radii in the range from $R(P_{i-1})_{\text{max}}$ to $R(P_i)_{\text{max}}$.

It is obvious that point $p \not\in SK_i$ is dominated by other points. Otherwise, it will be relayed to the base station and added to $SK_i$.

For a point $p \in SK_i$, we prove that point $p$ is not dominated by the other points by the following three cases.

Case 1. $p$ is not dominated by any point in $\bigcup_{j=1}^{i-1} P_j$. If there is a point $a \in \bigcup_{j=1}^{i-1} P_j$ dominating point $p$, there must be a point $q \in \bigcup_{j=1}^{i-1} SK_j$ that $q < a$ and $q < p$, or $q = a$, which contradicts the fact that $SK_i = \{p \mid p \in LSK(r), q \not\subset p, q \in \bigcup_{j=1}^{i-1} SK_j\}$.

Case 2. $p$ is not dominated by any point in $P_i$. Assume that point $p$ is relayed to sensor $v$ in the $i$th iteration. If there is a point $q$ at sensor $v$ with $q \in P_i$ and $q < p$, $p$ is impossible to be added to $LSK(v)$, and transmitted to the parent of $v$, which contradicts the fact that $p \in SK_i$.

Case 3. $p$ is not dominated by any point in $\bigcup_{j=i+1}^{k} P_j$. The range of radii of the points in subset $P_i$ is from $R(P_{i-1})_{\text{max}}$ to $R(P_i)_{\text{max}}$, the points in $\bigcup_{j=i+1}^{k} P_j$ thus have larger radii than point $p$. From Observation 1, it is obvious that $p$ cannot be dominated by any point in $\bigcup_{j=i+1}^{k} P_j$.

In summary, point $p \in SK_i$ is not dominated by any point in $P$. Therefore, $SK(P) = \bigcup_{i=1}^{k} SK_i$, delivered by algorithm a-FDP or g-FDP contains all the skyline points in $P$.

The following is the detailed description of algorithm a-FDP (algorithm g-FDP).

**Algorithm 2 Algorithm a-FDP (g-FDP) $(k, P, V, E)$**

begin
  generate arithmetic (geometric) series $R(P_i)_{\text{max}}, i \in [1, k]$;
  broadcast the series $R(P_i)_{\text{max}}, i \in [1, k]$ to the network;
  for $i = 1$ to $k$ do
    foreach sensor $v$ in the $i$th iteration do
      filter out the points at $v$ using $LF(v)_i$;
      if $v$ is a leaf sensor then
        calculate $LSK(v)$;
        send the points $\{p \mid p \in LSK(v), R(p) \leq R(P_i)_{\text{max}}\}$ to its parent;
      else
        receive the points from the children;
        calculate $LSK(v)$;
        send the points $\{p \mid p \in LSK(v), R(p) \leq R(P_i)_{\text{max}}\}$ to its parent;
      end
    end
    the base station $r$ calculates $LSK(r)$;
    $SK_i = \{p \mid p \in LSK(r), \forall q \in \bigcup_{j=1}^{i-1} SK_j, q \not\subset p\}$;
    Call algorithm $GSP(i, SK_i)$;
  end
end
Return $SK(P) = \bigcup_{i=1}^{k} SK_i$.

**IV. Dynamic Dataset Partition Algorithm**

Although the performance of algorithms a-FDP and g-FDP improve a lot, in comparison with that of algorithm Skyline Merge in terms of the total energy consumption and the maximum energy consumption among the sensors, they do suffer the following shortcomings.

For example, they take one extra preprocessing iteration on the routing tree to obtain the maximum and minimum radii of the points in $P$ without any skyline points delivered in that iteration. Also, the proposed arithmetic or geometric sequence of partition radius aims to filter out as many unlikely skyline points as possible from transmission in each iteration. However, in some datasets with skewed data distribution it is unavoidable that a disjoint subset may contain none of points. One such a scenario is, if the chosen value of $k$ is too large, $P$ is partitioned into many small disjoint subsets. Even if $P_i = \emptyset$, algorithm a-FDP or g-FDP still executes the $i$th iteration, which consumes energy without any gain. Furthermore, choosing an appropriate value of $k$ is difficult because the performance of algorithms a-FDP and g-FDP varies with different values of $k$ on different datasets, which will be shown in the later performance evaluation section. To overcome these mentioned shortcomings, in the following we propose an algorithm a-DDP (Dynamic Dataset Partition) which partitions the dataset dynamically.

**A. Dynamic Dataset Partition**

The number of iterations and the partition radius in algorithm DDP are determined dynamically by the data distribution of the dataset. The detail of algorithm DDP is depicted in the following.

Let $UR(v)_i$ be the upper bound of the transmission radius of sensor $v$ in the $i$th iteration. If sensor $v$ is a leaf sensor, $UR(v)_i$ is the maximum radius of all points forwarded by $v$. Otherwise, $UR(v)_i$ is calculated as follows. Assume that sensor $v$ has $d_v$ children, $u_1, \ldots, u_{d_v}$, and each child $u_j$ sends $Send(u_j)$ to sensor $v$, where $Send(u_j)$ is the set of points transmitted by sensor $u_j$ in the $i$th iteration, $1 \leq j \leq d_v$. Sensor $v$ calculates $LSK(v)_i$ that is the skyline of the points at sensor $v$ and the points received from its children. Then, $UR(v)_i$ is the minimum among the $d_v + 1$ maximum radii of the points sent by its children and the points in $LSK(v)_i$, i.e.,

$$UR(v)_i = \min\{R(\text{LSK}(v)_i)_{\text{max}}, R(\text{Send}(u_j))_{\text{max}}, 1 \leq j \leq d_v\},$$

where $R(\text{LSK}(v)_i)_{\text{max}}$ and $R(\text{Send}(u_j))_{\text{max}}$ are the maximum radii of the points in $\text{LSK}(v)_i$ and $\text{Send}(u_j)$, respectively. Obviously, $UR(v)_i \leq UR(u)_i$, when $u$ is a descendant sensor of $v$, and $UR(v)_i \leq UR(v)_i$ of any sensor $v$ in the sensor network, where $u$ is the base station. $UR(v)_i$ is also the partition radius of the $i$th partitioned subset $P_i$, i.e., the radii of all the points in $P_i$ are no greater than $UR(v)_i$. Therefore, the dataset is dynamically partitioned by $UR(v)_i$. Suppose that the points in each sensor $v$ are sorted in increasing order of their radii. Assuming that algorithm DDP performs the first $(i-1)$th iterations already, it now proceeds the $i$th iteration.

If sensor $v$ is a leaf sensor, it transmits all the points in $\text{LSK}(v)_i$ to its parent. Otherwise, sensor $v$ first calculates
that is the skyline of the points at \( v \) and the points received from its children, and then calculates upper bound of the transmission radius of sensor \( v \), \( UR(v)_i \). Sensor \( v \) finally transmits the points in \( LSK(v)_i \), whose radii are no greater than \( UR(v)_i \) to its parent.

Having received points from all of its children, the base station \( r \) calculates \( LSK(r)_i \) and \( UR(r)_i \). The newly found skyline points in the \( i \)th iteration is \( SK_i = \{ p \mid p \in LSK(r)_i, R(p) \leq UR(r)_i, \forall q \in \cup_{j=1}^{i} SK_j, q \neq p \} \). Several points in \( SK_i \) will be chosen for broadcast as the global filter to update the local filter at each sensor \( v \), i.e., \( LF(v)_{i+1} = LF(v)_i \cup GSF_i \), where \( GSF_i \) is the set of broadcast points in the \( i \)th iteration.

Clearly, the maximum number of iterations \( k \) is the height of the routing tree, because all the leaf sensors will transmit their points to their parents in the first iteration. However, transmitting all points at leaf sensors will incur the excessive energy consumption. Consider an extreme case, assume that all the points at a leaf sensor \( v \) are only dominated by a point at sensor \( u \). Let the base station \( r \) be the least common ancestor of sensors \( v \) and \( u \). This implies that all the points at sensor \( v \) will not be filtered out until they are relayed to the base station, which consumes much unnecessary energy. To this end, parameter \( \alpha \) is to limit the number of points transmitted by the leaf sensors in each iteration. For a leaf sensor \( v \), it only transmits the first \([[(\alpha \times |LSK(v)_i|)]\) points in \( LSK(v)_i \), to its parent in the \( i \)th iteration, where the points are sorted in increasing order of their radii, \( \alpha \) is a constant with \( 0 < \alpha \leq 1 \). If \( \alpha = 1 \), all the skyline points at the leaf sensors are transmitted. The algorithm partitioning the dataset dynamically with parameter \( \alpha \) is referred to as algorithm \( \alpha \)-DDP. The use of parameter \( \alpha \) can reduce the upper bound of the transmission radius of the leaf sensors, which also reduces the partition radius of each subset, thereby increasing the number of iterations \( k \). The more subsets are partitioned, the more unlikely skyline points will be filtered out by the global filter broadcast in previous subsets. Compared to its special case algorithm \( 1 \)-DDP where \( \alpha = 1 \), although algorithm \( \alpha \)-DDP takes more iterations, it reduces energy consumption from transmission, which will be shown in performance evaluation.

In case \( SK_i \) is empty, the algorithm \( Skyline \) Merge will be applied in a final iteration for finding the remaining skyline points. Algorithm \( \alpha \)-DDP terminates after the final iteration and the number of iterations \( k \) is then determined. The skyline of set \( P \) is \( SK(P) = \cup_{i=1}^{k} SK_i \).

### B. Correctness of Dynamical Dataset Partition Algorithm

The rest is to prove that the set \( SK(P) \) delivered by algorithm \( \alpha \)-DDP is the skyline on set \( P \) in the network.

**Theorem 2:** For a dataset \( P \), the skyline \( P \), \( SK(P) = \cup_{i=1}^{k} SK_i \), where \( SK_i \) is the new skyline points delivered by algorithm \( \alpha \)-DDP in the \( i \)th iteration.

**Proof:** Only the points with radii ranged from \( UR(r)_{i-1} \) to \( UR(r)_i \) are likely to be in \( SK_i \), which implies that the subset \( P_i \) of \( P \) contains the points with radii between \( UR(r)_{i-1} \) and \( UR(r)_i \).

Once \( SK_i = \emptyset \), this shows that there is not any skyline point with radii in the range between \( UR(r)_{i-1} \) and \( UR(r)_i \). Then, algorithm \( Skyline \) Merge is applied to return the skyline points with radii in the range between \( UR(r)_i \) and \( R(P)_{max} \), where \( R(P)_{max} \) is the maximum of the radii of the points in \( P \). Algorithm \( \alpha \)-DDP partitions the set \( P \) into \( k \) subsets and the range of the radii of the points in each subset \( P_i \) within \( (UR(r)_{i-1}, UR(r)_i] \) when \( 1 \leq i \leq k-1 \). The radii of the points in subset \( P_k \) ranges from \( UR(r)_{k-1} \) to \( R(P)_{max} \). The rest of argument is similar to the one in Theorem 1, omitted.

We conduct that \( SK(P) = \cup_{i=1}^{k} SK_i \) is the skyline on \( P \).

The following is the pseudocode of algorithm \( \alpha \)-DDP.

```plaintext
Algorithm 3 Algorithm \( \alpha \)-DDP(\( \alpha, P, V, E \))
begin
  \( i \leftarrow 1 \);
  repeat
    foreach sensor \( v \) in the \( i \)th iteration do
      filter out the points at \( v \), using \( LF(v)_i \);
      if \( v \) is a leaf sensor then
        calculate \( LSK(v)_i \);
        send the first \([[(\alpha \times |LSK(v)_i|)]\) points in \( LSK(v)_i \) to its parent;
      else
        receive the points from the children;
        calculate \( LSK(v)_i \) and \( UR(v)_i \);
        send the points \([p \mid p \in LSK(v)_i, R(p) \leq UR(v)_i]\) to its parent;
      end
    end
    the base station \( r \) calculates \( LSK(r)_i \);
    \( SK_i = \{ p \mid p \in LSK(r), q \neq p \} \);
    \( \alpha \)-DDP when \( 1 \leq i \leq k-1 \);
    Call algorithm \( GSF(i, SK_i) \);
    \( LF(v)_{i+1} = LF(v)_i \cup GSF_i \);
  end
  \( i \leftarrow i + 1 \);
  until \( SK_i = \emptyset \);
  obtain \( SK_{i+1} \) by applying algorithm \( Skyline \) Merge on the set of unexamined points;
  Return \( SK(P) = \cup_{i=1}^{k} SK_i \);
end
```

### V. PERFORMANCE EVALUATION

In this section we evaluate the performance of the proposed algorithms against existing algorithms in terms of the total energy consumption and the maximum energy consumption among the sensors.

#### A. Experimental Setting

We assume that the sensor network is used to monitor a 100m \( \times \) 100m region of interest. Within the region, 500 sensors are randomly deployed by the \( \text{NS} \)-2 simulator [15] and the base station is located at the square center. There is a communication channel between two sensors if they are within the transmission range of each other. We further assume that all sensors have the same transmission range (10 meters in this paper). As mentioned, the energy overhead on communication dominates the various energy consumption of a sensor and
we only consider the energy consumption on wireless communication in our experiments. Each \(d\)-dimensional point is represented by \(4 \times d\) bytes. It is supposed that the energy overhead on transmitting and receiving a header and the handshaking are \(p_t = 0.4608\) \(mJ\) and \(p_r = 0.1152\) \(mJ\). The energy consumption of transmitting and receiving one byte are \(R = 0.0144\) \(mJ\) and \(r_e = 0.00576\) \(mJ\), respectively, following the parameters given in a commercial sensor MICA2 [4]. In our experiments, we use the synthetic datasets with random and anti-correlated distributions. In each dataset \(10^6\) points are generated following the distribution and each sensor is assigned 2000 points randomly. We also use the real sensing dataset obtained by the Intel Lab at UC Berkeley [27], which is a data collection of 54 sensors. To assign the data to a network of 500 sensors in our setting, we partition the sensing sequence generated by each sensor into 10 consecutive segments and assign each segment to a sensor in our experiments. The data consists of temperature, humidity, light and voltage traces together as the 4-dimensional dataset. We use any 2 or 3 dimensional combinations of this 4-dimensional dataset to generate 2 and 3 dimensional datasets.

B. The Choice of Parameters

We first evaluate the performance of algorithms \(\alpha\)-FDP, \(g\)-FDP and \(\alpha\)-DDP with different values of \(k\) and \(\alpha\), on random datasets with dimensionality from 2 to 4.

Fig. 2(a) shows the curves of the ratios of the total energy consumption by algorithms \(\alpha\)-FDP and \(g\)-FDP to that by algorithm Skyline Merge with \(k\) from 2 to 6, while the ratios of the maximum energy consumption among the sensors by algorithms \(\alpha\)-FDP and \(g\)-FDP to that by algorithm Skyline Merge are illustrated in Fig. 2(b). It can be seen that algorithm \(g\)-FDP outperforms algorithm \(\alpha\)-FDP in terms of the total energy consumption and the maximum energy consumption among the sensors except the total energy consumption on 2-dimensional random datasets with \(k > 4\).

Fig. 2(a)-(b) also shows that the performance of algorithms \(\alpha\)-FDP and \(g\)-FDP varies with different values \(k\). For example, the ratio of the maximum energy consumption among the sensors by algorithm \(g\)-FDP to that by algorithm Skyline Merge ranges from 0.85 to 1.4. This implies that the choice of \(k\) greatly affects the performance of the proposed algorithms and it is not easy to determine an appropriate \(k\) for all datasets.

Fig. 2(c)-(d) plots the curves of the ratios of the total energy consumption and the maximum energy consumption among the sensors by algorithm \(\alpha\)-DDP to those by algorithm 1-DDP \((\alpha = 1)\). It can be shown that algorithm \(\alpha\)-DDP exhibits better performance than algorithm 1-DDP in terms of the both metrics, which implies that the setting \(\alpha < 1\) results in the energy saving from transmission. The performance curves on different datasets all increases gently with the increase of \(\alpha\), which means that the impact of different \(\alpha\) is minor to the performance of algorithm \(\alpha\)-DDP. It is concluded that we can choose appropriate \(\alpha\) for different datasets. We set \(k = 3\) in algorithm \(g\)-FDP and \(\alpha = 0.05\) in algorithm \(\alpha\)-DDP in the following experiments because the performance of these values of parameters plays relatively better than other choices.

C. Performance Analysis of Various Algorithms

We then investigate the performance of the proposed algorithms \(g\)-FDP and \(\alpha\)-DDP against existing algorithms by varying the dimensionality \(d\) from 2 to 4. We refer to the dynamic filter algorithm by Huang et al [7] as algorithm DF, the single point filter algorithm and grid index filter algorithm by Xin et al [24] as algorithm TF and algorithm GI, the certificate filter algorithm by Liang et al [23] as algorithm Cerf, respectively.

Fig. 3(a)-(c) illustrate the curves of the ratios of the total energy consumption by various algorithms to that by algorithm Skyline Merge on synthetic datasets with random and anti-correlated distributions and real datasets with dimensionality from 2 to 4, while the ratios of the maximum energy consumption by various algorithms to that by algorithm Skyline Merge are illustrated in Fig. 3(d)-(f). It can be seen that although algorithm Cerf is the best among the existing algorithms, algorithms \(\alpha\)-DDP outperforms the other algorithms overall in different datasets especially in high dimensional datasets. For example, the total energy consumption and the maximum energy consumption among the sensors by algorithm \(\alpha\)-DDP on 4-dimensional random datasets are 36% and 53% of those by algorithm Skyline Merge, while the total energy consumption and the maximum energy consumption among the sensors of algorithm Cerf are 54% and 73% of those by algorithm Skyline Merge. On different 4 dimensional datasets, algorithm \(g\)-FDP also performs better than algorithm Cerf. Through the performance analysis of various evaluation algorithms, clearly algorithm \(\alpha\)-DDP can prolong the network lifetime significantly.

VI. CONCLUSIONS

In this paper, we have studied the problem of skyline query processing in WSNs. We devised two algorithms for evaluating skyline query, which find the skyline points progressively. The novel techniques behind are to partition the dataset into disjoint subsets, followed by returning the skyline points on each subsequent subset progressively, using some found skyline points so far as a filter to filter out those unlikely skyline points in the currently processing subset from transmission. We finally conduct extensive experiments by simulations on both synthetic and real datasets. The experimental results show that the proposed algorithms outperform existing algorithms significantly in network lifetime prolongation.

Acknowledgment. It is acknowledged that the work by the authors is fully funded by a research grant No:DP0449431 by Australian Research Council under its Discovery Schemes.

REFERENCES


The Network Simulator-ns2. \url{http://www.isi.edu/nsnam/ns}.


