Wireless Sensor Network, **2010, 2, 838-849**
doi:10.4236/wsn.2010.211101 Published Online November 2010 (http://www.SciRP.org/journal/wsn)

Efficient Pr-Skyline Query Processing and Optimization in Wireless Sensor Networks*

Jianzhong Li, Shuguang Xiong
Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
E-mail: *lijzh@hit.edu.cn*, *n2xiong@gmail.com*

Received July 4, 2010; *revised August* 19, 2010; *accepted September* 27, 2010

Abstract

As one of the commonly used queries in modern databases, skyline query has received extensive attention from database research community. The uncertainty of the data in wireless sensor networks makes the corresponding skyline uncertain and not unique. This paper investigates the Pr-Skyline problem, *i.e.*, how to compute the skyline with the highest existence probability in a computational and energy-efficient way. We formulate the problem and prove that it is NP-Complete and cannot be approximated in a given expression. However, the proposed algorithm SKY-SEARCH with pruning techniques can guarantee the computational efficiency given relatively large input size, while the filter-based distributed optimization strategy significantly reduces the transmission cost and the required storage space of the sensor nodes. Extensive experiments verify the efficiency and scalability of SKY-SEARCH and the distributed optimizing strategy.

Keywords: Wireless Sensor Network, Query Processing, Uncertain Data, Probabilistic Data, Skyline Query

1. Introduction

As the development of computer science and wireless communication technologies, wireless sensor networks (WSNs) become important data sources and have been widely used in many applications [1-3]. In the database view, a WSN can be regarded as a distributed database, and efficient query processing methods for various types of queries in WSN become a hot topic in research community [4-6].

Skyline query is one of the most common-used queries for modern database management systems (DBMS) in many applications such as sensor data monitoring and business planning, and it receives extensive concerns from database research community [7-9]. Recently, efficient skyline query processing and skyline maintenance in WSNs have been studied in [2,6]. In a WSN, a record r from each sensor can be regarded as a D-dimensional vector. If the value of vector u is no less than the value of vector v in each dimension, and $u \neq v$, we say that u *dominates* v. Traditional skyline query (deterministic skyline query) returns the skyline of a data set, *i.e.*, the set of vectors that cannot be dominated by any other vectors. Once the data set is given for skyline query, the domination relationship and the skyline are both determinate.

However, the data obtained by a WSN are often uncertain and probabilistic due to various reasons [3,11,12]. According to the possible world model [13-15], the skyline over uncertain data is not determinate, and any possible skyline has an existence probability. In such a case, people may ask that "what are the skylines of the data with existence probabilities greater than a given constant p?" or "what is the skyline with the largest existence probability?" In this paper, we study the problem related to these questions. For brevity, we denote the problem of computing the maximum existence probability of the skylines as the Pr-Skyline problem.

Although previous works study skyline query processing over uncertain data [8,9,16], the problems that they focus on are different from the Pr-Skyline problem. In [8], the algorithms aim to find out the reverse skylines that are determined by the query point. [9,16] are concerned about the probability for a record to be included in a specific skyline, but not the existence probability of the skyline. Hence, existing approaches cannot be used for solving the Pr-Skyline problem to the best of our knowledge. Furthermore, because the sensors often have limited energy, computing ability and storage space [1,4,5], efficient Pr-Skyline query

*This work is supported by the Key Program of the national Natural Science Foundation of China (NSFC) under Grant No.60533110, the National Grand Fundamental Research 973 Program of China under Grant No.2006CB303005, and NSFC-RGC of China under Grant No.60831160525.

in WSN requires saving communication cost, computation cost and storage cost of the sensors as much as possible [15].

In this paper, we give the formal definition of the Pr-Skyline problem, and show its *domination graph* representation. We prove that it is NP-Complete, and it cannot be approximated in polynomial time within a $\delta^{c\log N-1}$ factor (*N* is the number of records) for any $0<\delta<0.5$ and $c>0$, unless P = NP. However, the proposed algorithm SKY-SEARCH with multiple pruning strategies shows its high efficiency on average even when the input data size is relatively large. Furthermore, we propose distributed optimization strategies based on *filters* to reduce communication cost and storage cost of the sensors. Extensive simulations show that the SKY-SEARCH algorithm and distributed optimization strategies have high efficiency and good scalability under variant existence probability distributions.

The rest of the paper is organized as follows. Section 2 briefly reviews related works, and Section 3 gives the definition of the Pr-Skyline problem and the cost model, followed by the theoretical results of the hardness of the problem in Section 4. We propose the SKY-SEARCH algorithm in Section 5, and provide the distributed optimization strategy in Section 6. Section 7 shows the simulation results, and Section 8 concludes the paper and suggests possible future works.

2. Related Work

The problem of skyline computation in context databases is first introduced by Borzsonyi *et al.* [17]. The skyline queries over deterministic data can be divided into two categories: static skyline queries [7,10,17] and dynamic skyline queries [18,19]. For static skyline queries [17], each attribute value of a record is static. Hence the skyline is unique for a given database. For dynamic skyline queries [19], each attribute value is computed according to the query. Deng *et al.* study the multi-source skyline query problem, in which the value of a record is defined as the minimum length of the shortest paths to the multiple query points [19]. Dellis *et al.* introduce the concept of reverse skyline, whose result skylines contain a given query point [18].

The probabilistic models of the uncertain data fall into two categories: one is the possible world model [13,14] [20], and the other is the probability function model [21], in which the existence of a record is represented by a probability density function. Till now, the research for query processing over uncertain data mainly focuses on nearest neighbor (NN) problem [21], K-nearest neighbor (K-NN) problem [22], join operation [23], ranking operation [20], and top-K queries [24]. Recently, skyline query over uncertain data has received much attention [8] [9,16]. Lian *et al.* model two types of reverse skylines, and propose efficient pruning techniques to reduce the search space [8]. Pei *et al.* study the *p*-skyline problem and present two efficient algorithms [16]. Li *et al.* propose novel algorithms to maintain *p*-skylines in sliding windows of data streams [9].

Because the energy of a sensor node is limited, and a node often spends a considerable part of energy on communication [1,5], many distributed algorithms for query processing focus on reducing communication cost [6,10]. The distributed algorithm proposed by Liang *et al.* handles skyline query and skyline maintenance [10], however, it cannot apply to skyline queries over uncertain data.

3. Problem Statement and Cost Model

In this section, we give preliminaries on skyline query, and then describe the Pr-Skyline query and the problem, followed by the network model and the cost model.

3.1. Preliminaries

Given *D* bounded and totally ordered domains A_1, A_2, …, A_D, define data space $\Omega = A_1 \times A_2 \times ... \times A_D$ and every *D*-dimensional vector is in the data space. For arbitrary two vectors r_i and r_j in Ω, denote r_i = < r_i[1], r_i[2], ..., r_i[D] >, and r_j = < r_j[1], r_j[2], ..., r_j[D] >. If $r_i[k] \le r_j[k]$ for all $1 \le k \le D$, and $r_i \ne r_j$, then r_j *dominates* r_i ($r_j \succ r_i$), otherwise r_j does not dominate r_i ($r_j \not\succ r_i$).

Let set $V = \{r_1, r_2, \ldots, r_N\}$ that consists of *N* vectors, the skyline of *V* is defined as SKY(*V*) = $\{r \in V | \forall w \in V, w \not\succ r\}$. In other words, SKY(*V*) is a subset of *V*, and each vector in SKY(*V*) cannot be dominated by any other vectors in *V*. For convenience, we use "vector" and "record" to represent a data item interchangeably.

3.2. The Pr-Skyline Query

An uncertain record in *V* consists of a vector r_i and its existence probability Pr$\{r_i\}$ ($0 <$ Pr$\{r_i\} \le 1$). We also denote the uncertain record as r_i. Clearly, the probability that r_i does not exist is 1 – Pr$\{r_i\}$.

Note that an uncertain data set *V* has multiple skylines in this case, and we denote the set of skylines of *V* as $\mathbb{S}(V)$ = SKY$_1$(*V*), SKY$_2$(*V*), …, SKY$_t$(*V*)}, in which the existence probability of each SKY$_i$(*V*) is defined as

$$\Pr\{\mathrm{SKY}_i(V)\} = \prod_{r\in \mathrm{SKY}_i(V)} \Pr\{r\} \prod_{r\in \mathrm{EXC}_i(V)} (1-\Pr\{r\}) \quad (1)$$

where EXC$_i$(*V*) = $\{r | r \in V, \forall e \in \mathrm{SKY}_i(V), e \not\succ r\}$. In other words, Pr{SKY$_i$(*V*)} is the product of two factors: one is the product of Pr$\{r\}$, $\forall r \in$ SKY$_i$(*V*), and the other one is the product of (1 – Pr{r}), $\forall$ r that cannot be dominated by any vector in SKY$_i$(V). The problem studied in this paper is to find out the skyline SKY$_{max}$ $\in \mathbb{S}(V)$ with the maximum existence probability, and we regard the query for SKY$_{max}$ as Pr-Skyline query.

3.3. The Cost Model

A WSN consists of n stationary sensor nodes s_1, s_2, ..., s_n, and each sensor node has N/n D-dimensional uncertain records. The sensor nodes are randomly deployed in a square area with side length L. We assume that the network is connected, and the sink locating in the bottom-left corner has infinite energy. Additionally, suppose the network is capable of in-network execution, and data are transmitted from a sensor node to sink via the path on a spanning tree of the network.

The energy cost in the query processing procedure consists of the communication cost and the computation cost of the sensor nodes. Because the communication cost for transmitting one bit by radio is typically no less than the computation cost for executing 1,000 CPU instructions [1], we can consider the communication cost as the energy cost when the time complexity of the algorithm running on each sensor node is relatively low, *i.e.*, linear to the data size.

According to [25], the energy cost for transmitting a data packet can be estimated as $E_p = E_1 + xE_2$, where E_1 is the fixed part of the energy, E_2 is the energy consumption per byte, and x is the number of bytes transmitted. Since a data packet accommodates λ bytes where λ is a constant, and most data packets are filled up with λ bytes for energy saving in skyline query processing, the communication cost can be estimated by counting the number of delivered data packets.

The communication cost for delivering a packet from sensor node s_i to s_j can be estimated as $h_{ij}E_p$, where h_{ij} is the number of hops from s_i to s_j. Suppose the distance between the two sensors is d_{ij}, h_{ij} is linear to d_{ij} as the equation $h_{ij} = \alpha d_{ij}$ shows, in which the coefficient α is a constant relying on routing mechanisms, e.g., to choose direct or hop-by-hop transmission in the communication range [6]. Therefore, the energy cost for delivering a data packet from s_i to s_j with distance d_{ij} can be estimated as

$$E_{ij} = \alpha \cdot d_{ij} \cdot E_p \qquad (2)$$

4. Hardness of the Pr-Skyline Problem

This section proves that the Pr-Skyline problem is NP-Complete, and it cannot be approximated in polynomial time within a $\delta^{c\log N-1}$ factor where N is the number of records for any constants $0 < \delta < 0.5$ and $c > 0$, unless P = NP. Before the proofs, we first introduce the *domination graph* representation of the problem.

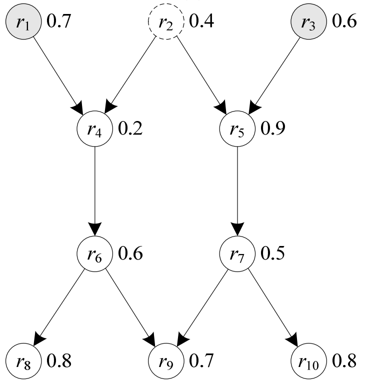
4.1. Domination Graph Representation

A domination graph G induced by the data set V has N vertices, and each record in V corresponds to a vertex in G. $\forall u, v \in V$, if $u \succ v$, there is a directed path from u to v in G, and u can reach v. Otherwise there is no directed path from u to v, and we say that u cannot reach v. Because the domination relation is transitive, *i.e.*, $r_i \succ r_k$ if $r_i \succ r_j$ and $r_j \succ r_k$, the Pr-Skyline problem is *equivalent* to the problem to find out a subset SKY(V) of the vertices in G with the condition that $\forall u, v \in \text{SKY}(V)$, u cannot reach v, such that Pr{SKY(V)} is maximized.

Note that data set V may induce more than one domination graphs. We call the domination graph with the minimum number of edges as the minimum domination graph of V, denoted as $G_M(V)$. It is easy to see that $G_M(V)$ is a directed acyclic graph (DAG), and there is no directed path with length > 1 from u to v for each edge uv in $G_M(V)$. In the following parts of the paper, we focus on the equivalent problem on $G_M(V)$. **Figure 1** shows a data

r_1	500	500	0.7
r_2	600	400	0.4
r_3	700	300	0.6
r_4	450	350	0.2
r_5	550	250	0.9
r_6	400	325	0.6
r_7	525	200	0.5
r_8	50	310	0.8
r_9	100	100	0.7
r_{10}	510	50	0.8

(a)



(b)

Figure 1. Sample uncertain data with the induced minimum domination graph. (a) The data set V is composed of ten 2-dimensional records with existence probabilities. (b) $G_M(V)$, in which each vertex indicates one record, and each directed edge indicates a dominating relation.

set with 10 records and its minimum domination graph.

4.2. NP-Completeness

Now we prove that the decision format of the Pr-Skyline problem is NP-Complete, *i.e.*, given data set $V = \{r_1, r_2, ..., r_N\}$ and threshold T, to determine whether there is a skyline SKY(V) of V, such that Pr{SKY(V)} ≥ T.

Theorem 1. *The Pr-Skyline problem is NP-Complete.*

Proof: First, we can find out a skyline of V, and compare its existence probability with T. Obviously, the procedure can finish in polynomial time with respect to N, which means that a solution to this problem can be verified in polynomial time, and the Pr-Skyline problem is in NP.

Then we construct a polynomial-time reduction from the Minimum Set Cover (MSC) problem to the Pr-Skyline problem. The MSC problem can be described as: given set S, its m subsets, and integer K ($K \le m$), determine whether there are K of the given m subsets, such that each item in S appears at least once in the K subsets. For arbitrary instance of the MSC problem, denote the n elements in S as $e_1, e_2, \ldots, e_n$, and denote the m subsets as $S_1, S_2, \ldots, S_m$, we construct an instance of the Pr-Skyline problem as follows. The domination graph G has $m+n(\lceil m\log_{1-\varepsilon}\varepsilon\rceil+1)$ vertices, in which m vertices refer to the m subsets of S, and each $\lceil m\log_{1-\varepsilon}\varepsilon\rceil+1$ vertices of the rest vertices refers to an element in S. If $e_j \in S_i$, we add a directed edge from the vertex of S_i to that of e_j. Finally, let the existence probability of each vertex be a constant $\varepsilon < 0.5$, and let $T = \varepsilon^K(1-\varepsilon)^{m-K}$. An example is illustrated in **Figure 2**. Because the size of graph G is polynomial with respect to $m \cdot n$, the construction only requires polynomial time.

If there are K subsets that cover all the elements in S for the MSC problem, then these K vertices are selected as the skyline of graph G. Because the K vertices dominate all the vertices in e_i ($1 \le i \le n$), and do not dominate the rest $m - K$ vertices, the existence probability of the skyline is $\varepsilon^K(1-\varepsilon)^{m-K} = T$. Hence this skyline is a result to the Pr-Skyline problem.

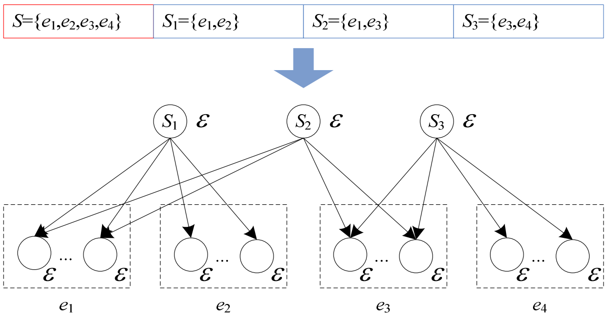


Figure 2. Construct an instance of Pr-Skyline from an instance of MSC. Each e_i refers to $\lceil m\log_{1-\varepsilon}\varepsilon\rceil+1$ vertices.

Conversely, if there is a skyline SKY(V) such that Pr{SKY(V)} ≥ T, then all the vertices in e_i ($1 \le i \le n$) are dominated by the vertices in SKY(V). To see this, suppose that a vertex in some e_i cannot be dominated by the vertices in SKY(V), it is clear that all the vertices in e_i cannot be dominated by the vertices in SKY(V). Let $A = e_i \cap$SKY(V) and $B = e_i \backslash$SKY(V), we have $\Pr\{\text{SKY}(V)\} \le \prod_{t\in A}\Pr\{t\}\prod_{t\in B}(1-\Pr\{t\}) = \prod_{t\in A}\varepsilon\prod_{t\in B}(1-\varepsilon)$. Since $\varepsilon < 0.5$, $\Pr\{\text{SKY}(V)\} \le (1-\varepsilon)^{\lceil m\log_{1-\varepsilon}\varepsilon\rceil+1} < \varepsilon^m$.

But $\Pr\{\text{SKY}(V)\} \ge T = \varepsilon^K(1-\varepsilon)^{m-K} \ge \varepsilon^m$, a contradiction. Therefore, all the vertices in e_i ($1 \le i \le n$) are dominated by the vertices in SKY(V). Suppose there are x vertices of the subsets in the skyline, because $\Pr\{\text{SKY}(V)\} = \varepsilon^x(1-\varepsilon)^{m-x} \ge T = \varepsilon^K(1-\varepsilon)^{m-K}$ and, $\varepsilon < 0.5$, we have $x \le K$. Hence there must be $K \ge x$ subsets covering all the elements in S.

Because the MSC problem is NP-Complete [26], the Pr-Skyline problem is NP-Complete.

4.3. Property

Lemma 1 (Raz and Safra [27]). $\forall c > 0$, *the MSC problem cannot be approximated within a* $c\log N$ *factor in polynomial time unless P = NP, in which N is the size of the set to be covered.*

Theorem 2. $\forall 0 < \delta < 0.5$, $c > 0$, *the Pr-Skyline problem cannot be approximated within a* $\delta^{c\log N-1}$ *factor in polynomial time unless P = NP, in which N is the number of data items.*

Proof: Denote the qualities associated with an optimal solution and an approximate solution for the MSC problem as OPT(MSC) and APP(MSC), respectively, and denote those for the Pr-Skyline problem as OPT(P-SKY) and APP(P-SKY), respectively. According to Lemma 2, APP(MSC) ≥ clog$N \cdot$OPT(MSC). Hence, approximation algorithms for the instances of Pr-Skyline problem that can be reduced from the MSC problem have the following ratio bound.

$$\begin{aligned}\frac{\text{APP(P-SKY)}}{\text{OPT(P-SKY)}} &= \frac{\varepsilon^{\text{APP(MSC)}}(1-\varepsilon)^{m\text{-APP(MSC)}}}{\varepsilon^{\text{OPT(MSC)}}(1-\varepsilon)^{m\text{-OPT(MSC)}}}\\ &\le \frac{\varepsilon^{c\log N\cdot\text{OPT(MSC)}}(1-\varepsilon)^{m\text{-}c\log N\cdot\text{OPT(MSC)}}}{\varepsilon^{\text{OPT(MSC)}}(1-\varepsilon)^{m\text{-OPT(MSC)}}} \qquad (3)\\ &= \left(\frac{\varepsilon}{1-\varepsilon}\right)^{(c\log N-1)\text{OPT(MSC)}}\end{aligned}$$

Let $\delta = \varepsilon/(1-\varepsilon)$, $0 < \delta < 1$ since $0 < \varepsilon < 0.5$. Because OPT(MSC) ≥ 1, APP(P-SKY) / OPT(P-SKY) $\leq \delta^{c\log N-1}$. Therefore, there is no polynomial-time aproximation algorithm with ratio bound more than $\delta^{c\log N-1}$ unless P = NP.

Theorem 2 suggests that polynomial-time algorithms for the Pr-Skyline problem can hardly obtain good approximation ratio bounds when the data set is large. As an alternation, this paper seeks for optimal solutions relying on search and pruning strategies.

5. The SKY-SEARCH Algorithm

The proposed SKY-SEARCH algorithm performs a search in an optimized order on the solution space to obtain the optimal skyline. In the search, several pruning methods are adopted to accelerate the procedure. SKY-SEARCH performs computing on the sink, and all the data of the sensor nodes need to be sent to the sink. This section describes the algorithm and analyzes its energy efficiency.

5.1. Algorithm Description

To compute the skyline with the maximum existence probability for the given *N* records, a straightforward approach is to enumerate all the possible skylines and then find out the optimal one with the maximum probability. For each skyline which is a subset of the records, it requires $O(N^2)$ time to determine whether the skyline is valid, and $O(N)$ time to compute its existence probability. Since there are 2^N possible skylines, this straightforward algorithm takes $O(N^2 \cdot 2^N)$ running time. Nevertheless, the algorithm wastes lots of time in computing invalid skylines.

The SKY-SEARCH algorithm improves the search order to avoid computing the invalid skylines. The basic idea is to maintain an *active set* to guide the search, which consists of the candidates that are possibly included in some skyline. At the initial state, the active set is initialized as the determined skyline regardless of the existence probabilities of the records. For example, the initial active set is $\{r_1, r_2, r_3\}$ in **Figure 1(b)**. In each step of the search, each valid skyline with its existence probability is computed by enumerating the status of the candidates in the active set, *i.e.*, selected as part of the skyline or not. The algorithm returns SKY_{max} with the maximum existence probability of all the valid skylines. Record *r* can be put into the active set if and only if its *dominated number* is 0, which is the number of records that can dominate *r*. If *r* is not selected in a skyline, the dominated numbers of the records that are dominated by *r* decrease by 1.

Algorithm 1. The SKY-SEARCH Algorithm
Input: *N D*-dimensional records $r_1, r_2, \ldots, r_N$
Output: The skyline SKY_{max} and its existence probability $\Pr\{\text{SKY}_{max}\}$

```
1:  initialize $\Pr\{\text{SKY}_{max}\} = 0.0$, and set $active = \varnothing$, $temp = \varnothing$;
2:  for $i = 0$ to $N-1$ do
3:          compute $r_i.d$, the number of records that dominate $r_i$;
4:          if $r_i.d == 0$ then
5:                  add $r_i$ into the active set;
6:  FindSkyline(1.0,0,the size of active);
7:  return $\text{SKY}_{max}$ and $\Pr\{\text{SKY}_{max}\}$;

Procedure FindSkyline(the existence probability cp of the current
skyline, the start position of current active set, the end position of
current active set)
8:  if $start \geq end$ then  ▷ the termination condition: $active = \varnothing$
9:          if $cp > \Pr\{\text{SKY}_{max}\}$ then  ▷ update current skyline
10:                 $\Pr\{\text{SKY}_{max}\} = cp$ and $\text{SKY}_{max} = temp$;
11:         return;
12: if $cp < \Pr\{\text{SKY}_{max}\}$ then  ▷ stop the search
13:         return;
14: put $active[start]$ into $temp$;
15: FindSkyline($cp * active[start].p$, $start + 1$, $end$);
16: remove $active[start]$ from $temp$;
17: if $cp * (1 - active[start].p) < \Pr\{\text{SKY}_{max}\}$ then
18:         return;
19: $add = 0$;  ▷ initialize the number of records to be added into active
20: for $i = 0$ to $N-1$ do
21:         if $active[start] \succ r_i$ and $active[start] \neq r_i$ then
22:                 $r_i.d--$;
23:                 if $r_i.d == 0$ then
24:                         put $r_i$ into the active set;
25:                         $add{+}{+}$;
26: FindSkyline($cp * (1 - active[start].p)$, $start + 1$, $end + add$);
27: for $i = 0$ to $N-1$ do
28:         if $active[start] \succ r_i$ and $active[start] \neq r_i$ then
29:                 $r_i.d{+}{+}$;
```

The pseudo code of the algorithm is shown in Algorithm 1, in which lines 1-5 initialize the dominated numbers, the active set, and the temporary skyline set \emph {temp}, line 6 calls FindSkyline to compute SKY_{max}. The FindSkyline procedure (1) computes current skyline and its probability if current active set is empty in lines 8-11, (2) searches the branch when the first item of current active set, *active*[*start*], is selected in the skyline in lines 14-16, and (3) searches the branch when *active*[*start*] is not selected in lines 19-29. Lines 12-13 and lines 17-18 are two simple pruning techniques (see the next subsection for details). Run the algorithm on the example shown in **Figure 1**, and the result $\text{SKY}_{max} = \{r_1, r_3\}$, and $\Pr\{\text{SKY}_{max}\} = 0.112$.

The SKY-SEARCH algorithm requires $O(N)$ storage space, and $O(N \cdot 2^N)$ time in the worst case (see **Figure 2**) because each step needs $O(N)$ time to re-compute the active set, and there are as large as $O(2^N)$ different active sets. Fortunately, we find efficient pruning techniques to dramatically reduce the running time on average, as illustrated in the next subsection.

5.2. Pruning Techniques

The following four pruning strategies can be used in the

SKY-SEARCH algorithm.

Pruning strategy 1. Stop recursion if the existence probability *cp* of the partially determined skyline in the active set is less than current $\Pr\{\mathrm{SKY}_{max}\}$. This is because any skyline that contains the partially determined part has its existence probability no more than *cp*, and less than $\Pr\{\mathrm{SKY}_{max}\}$.

The rest three pruning strategies are based on the following three theorems, respectively.

Theorem 3. *Denote I as the set of vertices that cannot be dominated by any other vertices in domination graph G,* $SKY_{max} \supseteq \{r|r \in I \text{ and } \Pr\{r\} > 0.5\}$.

Proof: Let $R=\{r|r\in I \text{ and } \Pr\{r\} > 0.5\}$, according to Equation (1),

$$\Pr\{\mathrm{SKY}_{max}\} = \prod_{r\in\mathrm{SKY}_{max}} \Pr\{r\} \prod_{r\in\mathrm{EXC}_{max}} (1-\Pr\{r\}),$$

in which $\mathrm{EXC}_{max} = \{r|r\in V \text{ and } \forall\, e\in\mathrm{SKY}_{max},\ e\nsucc r\}$. Suppose $\exists\, r$, $r\in R$ and $r\notin\mathrm{SKY}_{max}$. Let $\mathrm{SKY}' = \mathrm{SKY}_{\max}\cup\{r\}$, and $\mathrm{EXC}' = \{r|r\in V \text{ and } \forall\, e\in\mathrm{SKY}',\ e\nsucc r\}$, $\mathrm{EXC}_{\max} = \mathrm{EXC}'\cup W\cup\{r\}$, in which $W = \{e|e\in\mathrm{EXC}_{max}\backslash\{r\} \text{ and } r\succ e\}$. Because $\exists\, r\in V$, $0 < \Pr\{r\} \le 1$, we have

$$\frac{\Pr\{\mathrm{SKY}'\}}{\Pr\{\mathrm{SKY}_{max}\}} = \frac{\prod_{r\in\mathrm{SKY}'}\Pr\{r\}\prod_{r\in\mathrm{EXC}'}(1-\Pr\{r\})}{\prod_{r\in\mathrm{SKY}_{max}}\Pr\{r\}\prod_{r\in\mathrm{EXC}_{max}}(1-\Pr\{r\})} = \frac{\Pr\{r\}}{(1-\Pr\{r\})\prod_{e\in W}(1-\Pr\{e\})} > 1 \qquad (4)$$

which means that there is a skyline SKY' with $\Pr\{\mathrm{SKY}'\} > \Pr\{\mathrm{SKY}_{max}\}$, a contradiction. Hence, $r\in\mathrm{SKY}_{max}$ if $r\in I$ and $\Pr\{r\} > 0.5$.

Pruning strategy 2. If record *u* is dominated by a record in $R = \{r|r\in I, \Pr\{r\} > 0.5\}$, then *u* cannot be in SKY_{max} according to Theorem 3.

Theorem 4. *If the domination graph G is composed of k connected components G_1, G_2, ..., G_k, and there is no edge between any two connected components, then* $\Pr\{SKY_{max}\} = \prod_{i=1}^{k}\Pr\{\mathrm{SKY}(G_i)\}$.

Proof: Because there is no domination relation between any two connected components, the set $\mathrm{SKY} = \bigcup_{i=1}^{k}\mathrm{SKY}(G_i)$ is a skyline of *G*. Thus, $\Pr\{\mathrm{SKY}_{max}\} \ge \prod_{i=1}^{k}\Pr\{\mathrm{SKY}(G_i)\}$. Conversely, $\mathrm{SKY}_{max}\cap G_i$ must be a skyline of G_i for $1 \le i \le k$, hence $\Pr\{\mathrm{SKY}_{max}\} \le \prod_{i=1}^{k}\Pr\{\mathrm{SKY}(G_i)\}$. In summary, the equation holds.

Pruning strategy 3. Let G_u be the induced graph of the records that are not in current temporary set (the status of the records are not determined yet), compute the connected components of G_u and the optimal solution for each component, and then compute the global optimal solution by Theorem 4. This divide-and-conquer strategy avoids redundant computing of the domination relations, and improves the performance of the algorithm.

Theorem 5. *If the minimum domination graph G is a directed tree, in which there is only one edge pointing to each vertex except the root of the tree, the Pr-Skyline problem can be solved in polynomial time.*

Proof: We prove this theorem by giving a polynomial-time algorithm for this special case. Because the tree root *s* cannot be dominated by any other vertices, there are two cases for the skyline SKY_{max}: (1) *s* is in SKY_{max}, and $\Pr\{\mathrm{SKY}_{max}\} = \Pr\{s\}$. (2) *s* is not in SKY_{max}, and $\Pr\{\mathrm{SKY}_{max}\} = (1\text{-}\Pr\{s\}) \prod_{r\in\mathrm{Child}(s)} \mathrm{OPT}(r)$ according to Theorem 4, where Child(*s*) is the children set of *s*, and OPT(*r*) refers to the maximum existence probability of the skylines in the tree rooted at vertex *r*. Because computing OPT(*r*) is a sub-problem of the original problem, it can be solved by dynamic programming as the following equation:

$$\mathrm{OPT}(r) = \max\{\Pr\{r\}, (1-\Pr\{r\})\prod_{e\in\mathrm{Child}(r)}\mathrm{OPT}(e)\} \qquad (5)$$

To obtain OPT(*r*), OPT(*e*) for $\forall\, e\in\mathrm{Child}(r)$ should be computed in advance. Hence the algorithm should run in a bottom-up way, starting from the computation of the leaves, and ending with OPT(*s*), which is $\Pr\{\mathrm{SKY}_{max}\}$.

For arbitrary vertex *r* in the tree, it requires $O(|\mathrm{Child}(r)|)$ multiplies and *O*(1) comparisons to obtain OPT(*r*). Since there are *N* vertices in the tree, the time complexity of the algorithm is $O(N\cdot d)$, where *d* refers to the maximum size of the children sets of the vertices. Furthermore, it requires *O*(1) space to store OPT(*r*) $\forall\, r\in G$, the space complexity of the algorithm is *O*(*N*). Thus, we have presented a polynomial-time dynamic programming algorithm, which completes the proof of the theorem.

Pruning strategy 4. Find out the forest that consists of directed trees in the induced graph G_u, and compute the skylines of the trees by the above dynamic programming algorithm. If a directed tree is a single vertex *r*, the skyline on the tree with maximum existence probability $\max(\Pr\{r\},\ 1 - \Pr\{r\})$ can be immediately computed without enumerating the status of the vertex. Furthermore, this pruning strategy shows notable efficiency in dealing with high-dimensional data because the domination graph is sparser than with low-dimensional data, and there are probably more directed trees generated in the search.

5.3. Energy Efficiency and Workload Balance

Energy efficiency. Consider the rectangle area on the plane centered at (*x*, *y*) with the sink as the original point, and its side lengths dx and dy. If dx and dy are small enough, the distance from any sensor node in the area to the sink can be regarded as $\sqrt{x^2+y^2}$. Define the density of the sensor network as $\rho = N/L^2$, and the number of sensor nodes in the area can be estimated as ρ dxdy. According to the analysis in Section 3.3, the energy cost to deliver a data packet from a sensor node in the area to the sink is estimated as $\alpha\sqrt{x^2+y^2}$. Suppose a data packet can contain at most β records, since each sensor node has N/n records, the energy cost for all the sensor nodes in the area to send their data to the sink is estimated as $\left(N\alpha/\beta L^2\right)\sqrt{x^2+y^2}\mathrm{d}x\mathrm{d}y$. Therefore, the energy cost E_c of the sensor network can be estimated as the integral of the energy cost in unit area on the whole region, which is

$$E_c = \iint_\Gamma \frac{N\alpha}{\beta L^2}\sqrt{x^2+y^2}dxdy = \frac{1}{3}\left[\ln(\sqrt{2}+1)+\sqrt{2}\right]NL\alpha/\beta \tag{6}$$

in which the integral area Γ is the square region with side length L. Equation (6) indicates that the energy cost of the SKY-SEARCH algorithm is proportional to the product of data size N and network size L when the routing algorithm and packet size are both fixed.

Workload balance. Let the communication radius of the sensor nodes be r_c, the number of the sensor nodes less than r_c away from the sink is about $1/4\rho\pi r_c^2$. Because these sensor nodes have to forward N/β data packets generated by all the sensor nodes, the average number of forwarded data packets per sensor is $N/(1/4\rho\pi r_c^2) = 4NL^2/(n\beta\pi r_c^2)$. On the other hand, the sensor nodes at the edge of the network only need to send their own data to their neighbors, hence the workload of these sensors is $N/(n\beta)$. It is clear that in the centralized algorithm, the heaviest workloads of the sensors are $4L^2/(\pi r_c^2)$ times as much as the lightest workloads.

Besides, the required storage spaces of the sensor nodes are also unbalanced. Based on the above analysis, the sensor nodes within one hop to the sink have to store $4NL^2/(n\beta\pi r_c^2)$ data packets on average if the data cannot be sent to the sink in time and are stored on these sensor nodes, while the nodes at the edge of the network only need to store their own data, N/n packets.

6. Distributed Optimization Strategy

The distributed optimization strategy takes three steps. First, the sink s obtains I, the set of vertices that cannot be dominated by any other vertices, and $R = \{r|r\in I$ and $\Pr\{r\} > 0.5\}$. Then s broadcasts $\Pr\{I\}$ to obtain $\mathrm{SKY}_{filter} = R\cup\{r|\Pr\{r\} > 1 - \Pr\{I\}\}$. Finally, s broadcasts SKY_{filter} to obtain the data that cannot be dominated by it.

To obtain I and R, each node u needs to compute the skyline I_u of the tree rooted at u assuming all the existence probabilities of the records are equal to 1, and then uploads I_u to its parent. When the sink receives all the skylines from its children, it computes I and R, and then broadcasts R to each sensor node. When a sensor node receives R, it uploads the records which are generated by it and cannot be dominated by any records in R.

The optimization strategy also uses the existence probability of skyline I as a filter condition. Specifically, if $\exists r$, $\Pr\{r\} > 1 - \Pr\{I\}$, then any record dominated by r cannot be in SKY_{max}. To see this, suppose $\exists x$, $r \succ x$ and $x\in\mathrm{SKY}_{max}$, $r\notin\mathrm{SKY}_{max}$ and r cannot be dominated by any record in SKY_{max}, hence $\Pr\{\mathrm{SKY}_{max}\} \le 1 - \Pr\{r\}$. But we also have $\Pr\{I\} > 1 - \Pr\{r\}$ since I is a skyline, a contradiction. Thus, all the records dominated by r cannot be in SKY_{max}.

Based on the above analysis, the sink should also broadcast $\Pr\{I\}$ to all the sensor nodes. Let $U = \{r|\Pr\{r\} > 1 - \Pr\{I\}\}$, when a sensor node receives $\Pr\{I\}$, it uploads the records which are generated by itself and cannot be dominated by any record in U. Recall that the local data dominated by any record in R are not uploaded, we finally choose $R\cup U$ as the filter set.

6.1. Algorithm Description and Analysis

The computation cost of the distributed optimization algorithm consists of three parts: (1) the cost for computing the local skyline, (2) the cost for computing the local set of data with existence probability larger than $1 - \Pr\{I\}$, and (3) the cost for computing the local data that cannot be dominated by the partially obtained filter. For each sensor node u, the time complexities are $O(w^2)$, $O(w)$, and $O(w\cdot|\mathrm{SKY}_{filter}|)$, respectively, where w is the size of the data on u. Thus, u spends $O(w\cdot(w+|\mathrm{SKY}_{filter}|))$ time in the procedure. As for storage cost, because u has to store its local data and skyline, and SKY_{filter}, the required storage space is $O(w + |\mathrm{SKY}_{filter}|)$. In the following subsection, we discuss the size of SKY_{filter}.

The pseudo codes of the optimization algorithm running on the sink and the sensor nodes are shown in Algorithm 2 and Algorithm 3, respectively.

6.2. The Size of Skyline and SKY_{filter}

We first discuss the size of the skyline on deterministic data. Given N D-dimensional vectors $V = \{r_1, r_2, ..., r_N\}$, and $\Pr\{r_i \succ r_j\} = \Pr\{r_j \succ r_i\}$ for arbitrary r_i and r_j ($i \ne j$), we have $1 \le |\mathrm{SKY}(V)| \le N$. If there is one vector that can dominate the rest $N - 1$ vectors, then $|\mathrm{SKY}(V)| = 1$. If none of these vectors can be dominated by any other vector, then $|\mathrm{SKY}(V)| = N$. Because these vectors in set V are different from each other, $\mathrm{SKY}(V)$ cannot be empty,

Algorithm 2. The optimization algorithm on the sink

```
1: broadcast the request for Pr-Skyline query;
2: receive the skylines from the children;
3: compute Pr{$I$} and $R$ = {$r|r \in I$ and Pr{$r$} > 0.5}, then broadcast Pr{I};
4: receive the data from the children, and compute SKY$_{filter}$ = $R \cup U$;
5: broadcast SKY$_{filter}$;
6: receive the filtered data from the children as the input of SKY-SEARCH, and then compute SKY$_{max}$;
```

Algorithm 3. The optimization algorithm on the sensor nodes

```
1: receive and broadcast the request for Pr-Skyline query;
2: receive the skylines from the children;
3: compute local skyline and forward it to the parent;
4: receive and broadcast Pr{$I$} from the parent;
5: upload local records with existence probability larger than 1 – Pr{$I$}, and forward the data from the children to the parent;
6: receive and broadcast SKY$_{filter}$;
7: upload local records not dominated by SKY$_{filter}$ and forward the data from the children to the parent;
```

i.e., |SKY(V)| $\neq$ 0.

Theorem 6. *The expectation value of |SKY(V)| is*

$$E\left(|\,SKY(V)\,|\right)=\left[1-\sum_{i=1}^{N-1}(-1)^{i+1}C_{N-1}^{i}(i+1)^{-D}\right]N\ \cdot$$

proof: First, we prove that Pr{$r_i \in$SKY(V)} = $1-\sum_{i=1}^{N-1}(-1)^{i+1}C_{N-1}^{i}(i+1)^{-D}$. For arbitrary r_i, $r_j \in V$ ($i \neq j$), because Pr{$r_i[k] \geq r_j[k]$} = 2^{-1} for $1 \leq k \leq D$, and the values in one dimension are independent from those in another dimension, Pr{$r_i \succ r_j$} = 2^{-D}. Generally, the probability for k vectors to simultaneously dominate r_i is Pr{$r_{j1} \succ r_i, r_{j2} \succ r_i, \ldots, r_{jk} \succ r_i$} = k^{-D}.

If $r_i \in$SKY(V), then r_i cannot be dominated by the rest N – 1 vectors. As a consequence, Pr{$r_i \in$SKY(V)} = 1 – Pr{$r_1 \succ r_i$ OR ... OR $r_{i-1} \succ r_i$ OR $r_{i+1} \succ r_i$ ... OR $r_N \succ r_i$}. Because the events $r_j \succ r_i$ and $r_k \succ r_i$ (j, $k \neq i$) are independent, the equation can be rewritten as Pr{$r_i \in$SKY(V)} = 1 – [C_{N-1}^{1}Pr{r_i is dominated by the rest 1 vector} – C_{N-1}^{2}Pr{r_i is dominated by the rest 2 vectors} + ... + $(-1)^N C_{N-1}^{N-1}$Pr{r_i is dominated by the rest N – 1 vectors}] = $1-\sum_{i=1}^{N-1}(-1)^{i+1}C_{N-1}^{i}(i+1)^{-D}$. Because there are N vectors,

$$E\left(|\,SKY(V)\,|\right)=\left[1-\sum_{i=1}^{N-1}(-1)^{i+1}C_{N-1}^{i}(i+1)^{-D}\right]N$$

Theorem 6 indicates that the expected size of the skyline is far smaller than data size N, especially when D is small. **Figure 3** illustrates the expected size under different number of dimensions when N varies from 1 to 50.

Because R is the subset of some skyline on deterministic data, the expected size of R cannot be larger than |SKY(V)|. Moreover, the expected size of U is relative to the existence probabilities of the data. If most existence probabilities are near to 1, U may become very large. Because SKY$_{filter}$ = $R \cup U$, the communication cost for broadcasting SKY$_{filter}$ is larger than that for collecting all the data to the sink when $U > N/n$. In this case, it is a better choice to let SKY$_{filter}$ = R, regardless of U.

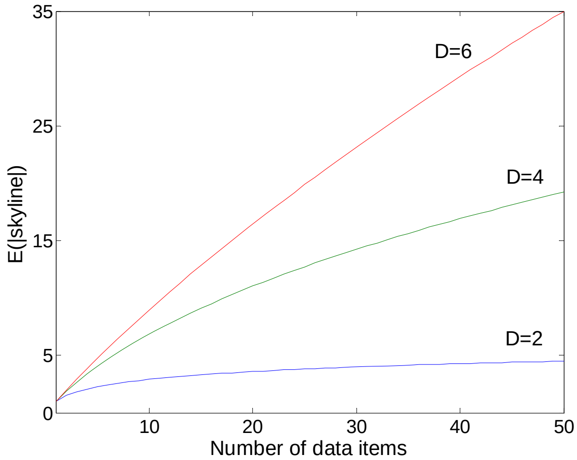


Figure 3. The relation between the expected size of skyline and the number of records with variant dimensions.

7. Simulations

The simulations consist of two parts: (1) the running time of the SKY-SEARCH algorithm, and (2) the energy cost of the distributed algorithm. The simulations run on randomly-generated network topologies in a 300 by 300 m^2 square area, and the communication radius varies from 50 m to 100 m. Each sensor node has 100 packets of data. The existence probabilities of the records are of three types: uniform distribution, normal distribution, and the derived distribution from Poisson distribution. A record in the derived distribution has an existence probability μ / x where x is the value of the random variable in Poisson distribution with $\lambda = 1$, and μ is the normalization coefficient. In the derived distribution, most records have existence probabilities near to 1. The simulations run on a PC with a Pentium 2.0 GHz CPU and 2 GB memory, and the network simulator is TOSSIM [28].

7.1. Running Time

When D = 2, the running time of the SKY-SEARCH algorithm with variant data size is shown in **Figure 4**, **Figure 5** and **Figure 6**. When the number of records varies from 10 to 10^5, the running time of the algorithm without pruning (denoted as SKY-SEARCH in the figures) rapidly increases from 0ms to 10^4 ms level, while the running time of the algorithm with pruning techniques (denoted as SKY-SEARCH-OPT in the figures) is of 1-2 orders of magnitude lower than the previous one.

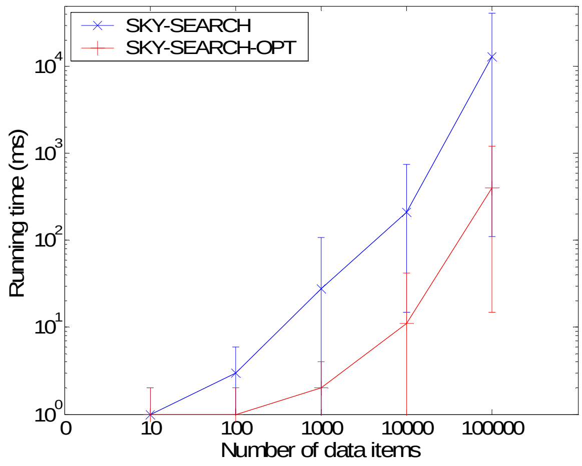


Figure 4. Running time with uniform distribution of the existence probabilities.

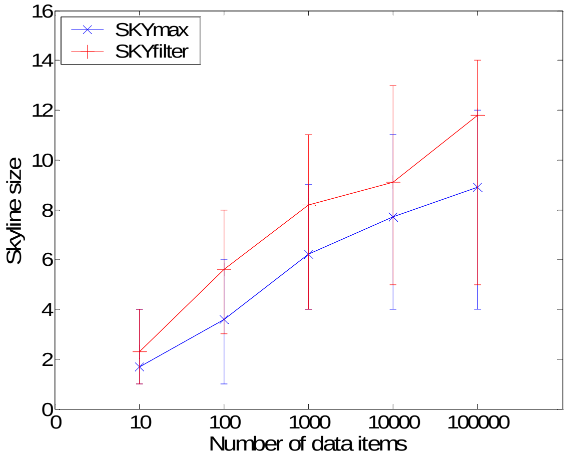


Figure 7. Impact of *D* with uniform distribution of the existence probabilities.

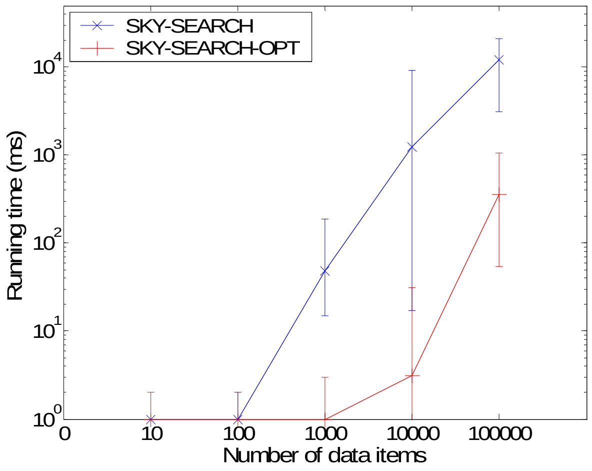


Figure 5. Running time with normal distribution of the existence probabilities.

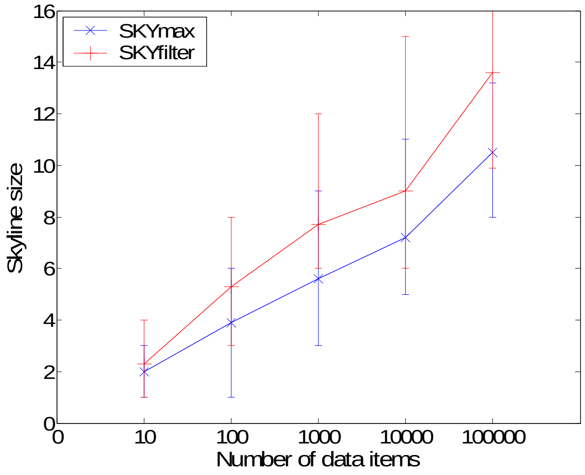


Figure 8. Impact of *D* with normal distribution of the existence probabilities.

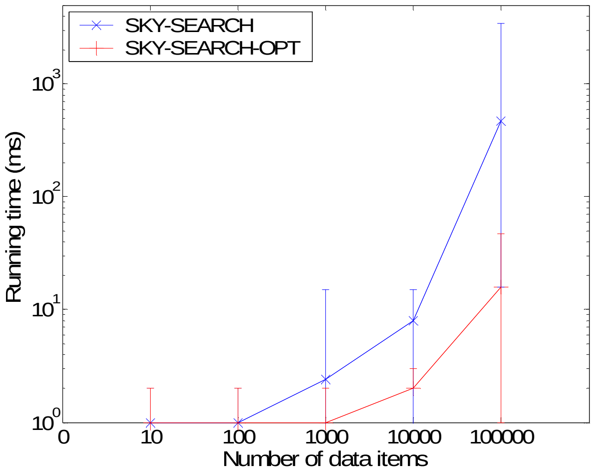


Figure 6. Running time with the derived distribution of the existence probabilities.

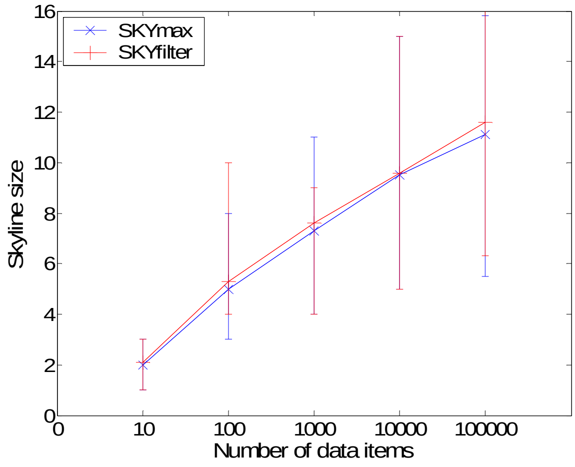


Figure 9. Impact of *D* with the derived distribution of the existence probabilities.

When $N = 200$, the running time of the algorithm with variant dimension D is illustrated in **Figure 7, Figure 8**, and **Figure 9**. As D increases from 2 to 4, the algorithm without pruning runs more than 10^8 ms (not shown in the figures), while the algorithm with pruning runs less than 10^4 ms. Besides, the running time of the algorithm with pruning appears to be a downward trend after rising for the first. The reason is two-folds. First, when D begins to increase from 2, the number of domination relations starts to decrease from a very dense situation, hence the search space increases. Second, when D increases to a large number, *i.e.*, 8, the number of domination relations is fairly small, and pruning strategy 4 shows efficiency.

Figure 10, Figure 11 and **Figure 12** depict the sizes of SKY_{max} and SKY_{filter} with variant data size when $D = 2$. The size of SKY_{max} grows from 2 to about 10 as the number of records grows from 10 to 10^5. $|SKY_{filter}|$ is always larger than $|SKY_{max}|$ with uniform distribution and normal distribution, while they are almost the same with the derived distribution. These results are consistent with the analysis in Section 6.

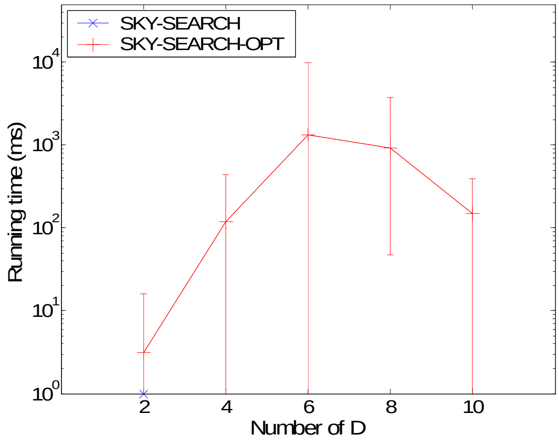


Figure 10. The sizes of SKY_{max} and SKY_{filter} with uniform distribution of the existence prob.

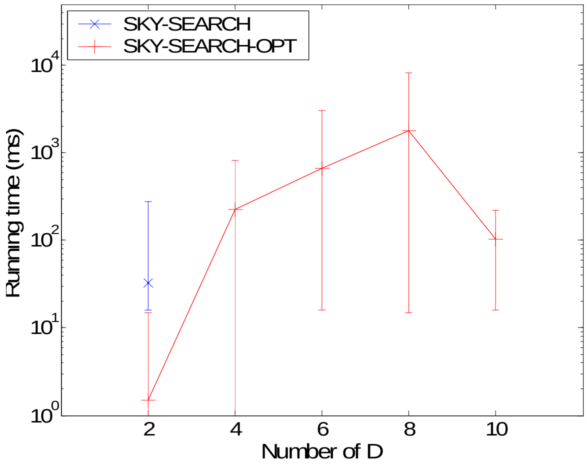


Figure 11. The sizes of SKY_{max} and SKY_{filter} with normal distribution of the existence prob.

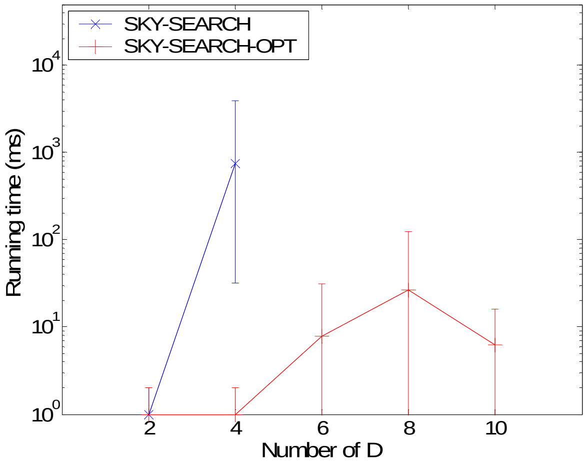


Figure 12. The sizes of SKY_{max} and SKY_{filter} with the derived distrib. of the existence prob.

7.2. Energy Cost

The workloads of the sensor nodes of a 100-node network are illustrated in **Figure 13**, **Figure 14**, and **Figure 15**. The workloads of a sensor node u in both algorithms are sorted by the number of packets sent by u without the optimization in descending order. For all of the three distributions, there are more than 20 nodes whose workloads are more than 100 packets, while all of the nodes forward less than 100 packets with the optimization.

Figure 16 illustrates the energy cost of the network as the network size grows from 20 nodes to 100 sensor nodes. The number of packets grows from about 5000 to more than 25000 when using the algorithm without optimization, and each node send more than 250 packets on average. With the optimization, the number of packets grows slowly (about 5000 packets when there are 100 sensor nodes). These results indicate that the optimization strategies notably reduce the communication cost.

Figure 17 and **Figure 18** show the energy costs for delivering SKY_{filter} and un-dominated data when $D = 2$ and $D = 4$, respectively. We can see that most packets are for uploading local skylines and broadcasting SKY_{filter} for all of the three distributions, while the cost for uploading un-dominated data takes a small partition. The effect of SKY_{filter} is obvious.

8. Conclusions and Future Work

This paper proposes an efficient algorithm SKY-SEARCH with distributed optimization strategies for the Pr-Skyline problem in WSNs. Although the problem is proved

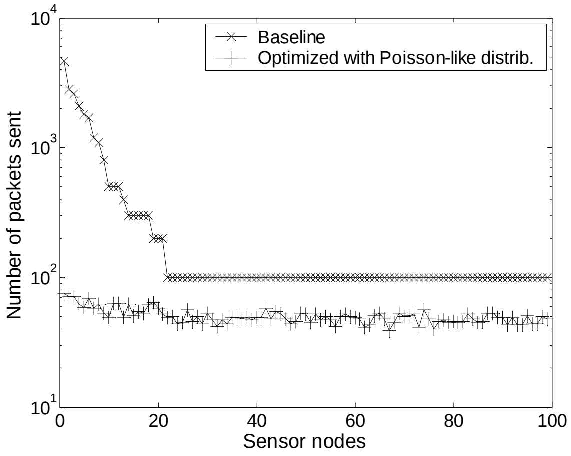


Figure 13. Workloads of the sensors with uniform distribution of the existence probabilities.

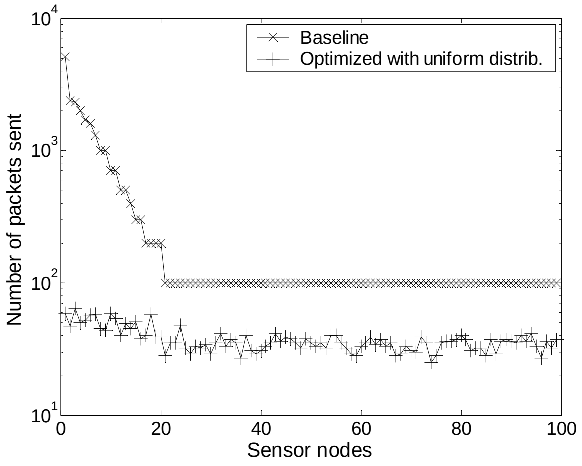


Figure 14. Workloads of the sensors with normal distribution of the existence probabilities.

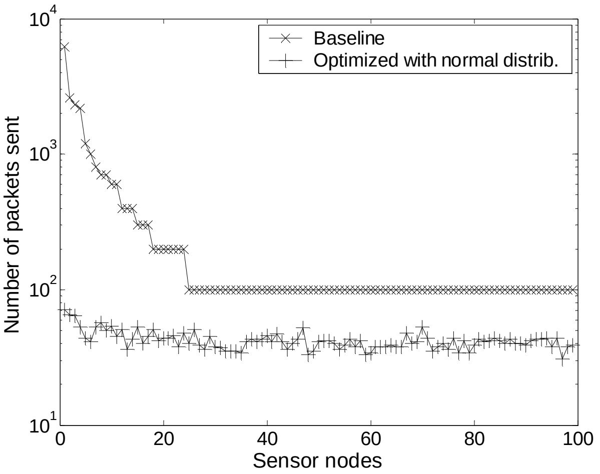


Figure 15. Workloads of the sensors with the derived distribution of the existence prob.

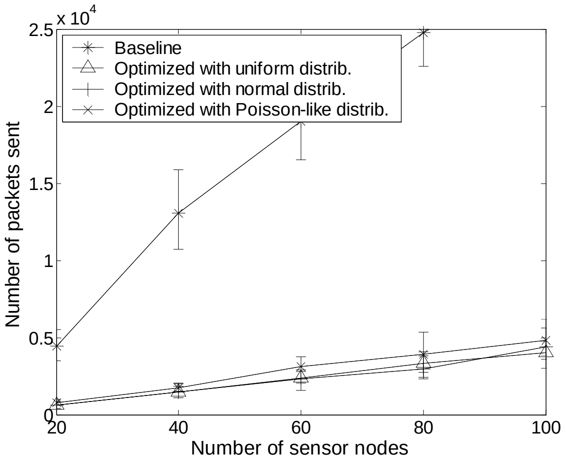


Figure 16. Energy cost of the network.

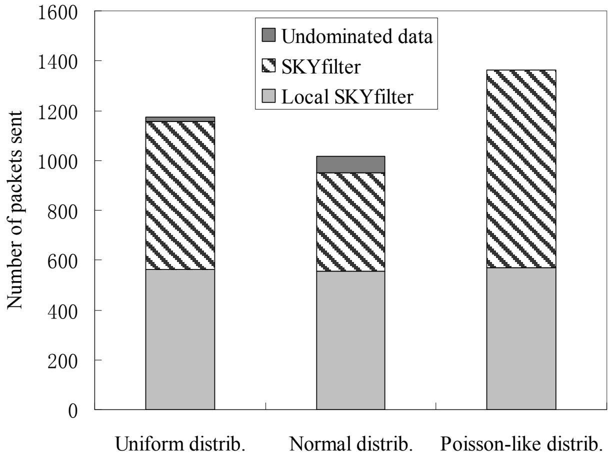


Figure 17. Energy costs for delivering variant types of data (D = 2).

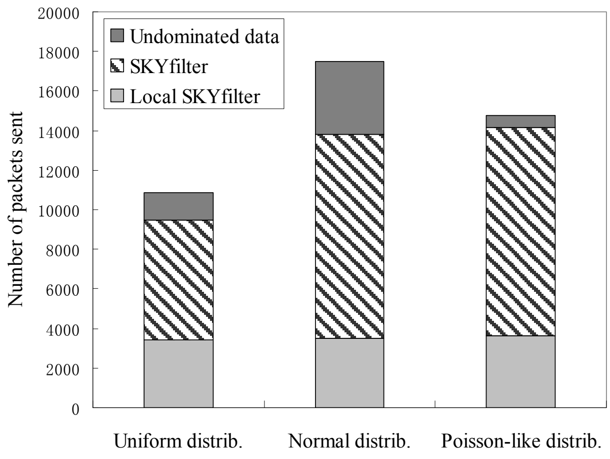


Figure 18. Energy costs for delivering variant types of data (D = 4).

as an NP-Complete problem, and cannot be approximated within a given expression, the algorithm with pruning techniques shows its efficiency given relatively large input size, and the filter-based distributed optimization strategy significantly reduce the transmission cost and the required storage space of the sensor nodes by extensive simulations.

In the future, we will consider how to maintain skylines for a time period over uncertain data streams since there are potential requirements for continuous skyline query. Besides, efficient algorithms for complex queries over uncertain data, *i.e.*, similarity search, is one of the suggested future works.

9. References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, *et al.*, "Wireless Sensor Networks: A Survey," *Computer Networks*, Vol. 38, No. 4, 2002, pp. 393-422.

[2] W. Liang, B. Chen and J. Yu, "Energy-Efficient Skyline Query Processing and Maintenance in Sensor Networks," in *Proceedings of ACM CIKM*, New York, 2008, pp. 1471-1472.

[3] A. Deshpande, C. Guestrin, S. Madden, *et al.*, "Model-Driven Data Acquisition in Sensor Networks," in *Proceedings of VLDB*, New York, 2004, pp. 588-599.

[4] N. Shrivastava, C. Buragohain, D. Agrawal, *et al.*, "Medians and Beyond: New Aggregation Techniques for Sensor Networks," in *Proceedings of SenSys*, ACM, New York, 2004, pp. 239-249.

[5] S. Madden, M. Franklin, J. Hellerstein, *et al.*, "TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks," in *Proceedings of OSDI*, ACM, New York, 2002, pp. 131 -146.

[6] X. Yang, H. B. Lim, M. Ozsu, *et al.*, "In-Network Execution of Monitoring Queries in Sensor Networks," in *Proceedings of ACM SIGMOD*, ACM, New York, 2007, pp. 521-532.

[7] A. Vlachou, C. Doulkeridis and Y. Kotidis, "Angle -Based Space Partitioning for Efficient Parallel Skyline Computation," in *Proceedings of SIGMOD*, ACM, New York, 2008, pp. 227-238.

[8] X. Lian and L. Chen, "Monochromatic and Bichromatic Reverse Skyline Search over Uncertain Databases," in *Proceedings of ACM SIGMOD*, ACM, New York, 2008, pp. 213-226.

[9] J. Li, S. Sun and Y. Zhu, "Efficient Maintaining of Skyline over Probabilistic Data Stream," in *Proceedings of IEEE ICNC*, Washington D.C., 2008, pp. 378-382.

[10] W. Liang, B. Chen and J. Yu, "Energy-Efficent Skyline Query Processing and Maintenance in Sensor Networks," in *Proceedings of ACM CIKM*, ACM, New York, 2008, pp. 1471-1472.

[11] R. Cheng, Y. Xia, S. Prabhakar, *et al.*, "Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data," in *Proceedings of VLDB*, ACM, New York, 2004, pp. 876-887.

[12] R. Cheng, D. Kalashnikov and S. Prabhakar, "Evaluating Probabilistic Queries over Imprecise Data," in *Proceedings of ACM SIGMOD*, ACM, New York, 2003, pp. 551-562.

[13] C. Koch and D. Olteanu, "Conditioning Probabilistic Databases," in *Proceedings of VLDB*, ACM, New York, 2008, pp. 313-325.

[14] R. Cheng, J. Chen and X. Xie, "Cleaning Uncertain Data with Quality Guarantees," in *Proceedings of VLDB*, ACM, New York, 2008, pp. 722-735.

[15] R. Sarkar, X. Zhu and J. Gao, "Double Rulings for Information Brokerage in Sensor Networks," in *Proceedings of ACM MOBICOM*, ACM, New York, 2006, pp. 286- 297.

[16] J. Pei, B. Jiang, X. Lin, *et al.*, "Probabilistic Skylines on Uncertain Data," in *Proceedings of VLDB*, ACM, New York, 2007, pp. 15-26.

[17] S. Borzsonyi, D. Kossmann and K. Stocker, "The Skyline Operator," in *Proceedings of IEEE ICDE*, Washington D.C., 2001, pp. 421-430.

[18] E. Dellis and B. Seeger, "Efficient Computation of Reverse Skyline Queries," in *Proceedings of VLDB*, ACM, New York, 2007, pp. 291-302.

[19] K. Deng, X. Zhou and H. Shen, "Multi-Source Skyline Query Processing in Road Networks," in *Proceedings of IEEE ICDE*, Washington D.C., 2007, pp. 796-805.

[20] M. Hua, J. Pei, W. Zhang, *et al.*, "Ranking Queries on Uncertain Data: A Probabilistic Threshold Approach," in *Proceedings of ACM SIGMOD*, ACM, New York, 2008, pp. 673-686.

[21] R. Cheng, J. Chen and M. Mokbel, "Probabilistic Verifiers: Evaluating Constrained Nearest-Neighbor Queries over Uncertain Data," in *Proceedings of IEEE ICDE*, Washington D.C., 2008, pp. 973-982.

[22] G. Beskales, M. Soliman, I. Ilyas, "Efficient Search for the Topk Probable Nearest Neighbors in Uncertain Databases," in *Proceedings of VLDB*, ACM, New York, 2008, pp. 326-339.

[23] R. Cheng, S. Singh, P. Prabhakar, *et al.*, "Efficient Join Processing over Uncertain Data," in *Proceedings of ACM CIKM*, ACM, New York, 2008, pp. 738-747.

[24] C. Jin, K. Yi, L. Chen, *et al.*, "Sliding-Window Top-k Queries on Uncertain Streams," in *Proceedings of VLDB*, ACM, New York, 2008, pp. 301-312.

[25] A. Silberstein, R. Braynard, C. Ellis, K. Munagala and J. Yang, "A Sampling-Based Approach to Optimizing Top-k Queries in Sensor Networks," in *Proceedings of IEEE ICDE*, Washington D.C., 2006, pp. 68-77.

[26] M. Garey and D. Johnson, "Computers and Intractibility: A Guide to the Theory of NP-Completeness," Bell Telephone Laboratories, Inc, 1979.

[27] R. Raz and S. Safra, "A Sub-Constant Error-Probability Low-Degree Test, and Sub-Constant Error-Probability PCP Characterization of NP," in *Proceedings of ACM STOC*, ACM, New York, 1997, pp. 475-484.

[28] TOSSIM: A Simulator for TinyOS Networks, [EB/OL] http://www.cs.berkeley.edu/pal/pubs/nido.pdf