Poster Paper

Top-*k* Skyline Groups Queries

Haoyang Zhu*, Peidong Zhu*, Xiaoyong Li*†, Qiang Liu*
* College of Computer,
† Academy of Ocean Science and Engineering,
National University of Defense Technology, Changsha, China
{zhuhaoyang, pdzhu, sayingxmu}@nudt.edu.cn, qiangl.ne@hotmail.com

ABSTRACT

The top-k skyline groups query (k-SGQ) returns k skyline groups that dominate the maximum number of points in a given data set. It combines the advantages of skyline groups and top-k queries. The k-SGQ is an important tool for queries that need to analyze not only *individual* points but also *groups* of points, and can be widely used in areas such as decision support applications, market analysis and recommendation system. In this paper, we formally define this new problem and design an efficient algorithm to solve this problem. Extensive experimental results show that our algorithm is effective and efficiency.

1. INTRODUCTION

The skyline query [1] is widely used in multi-criteria optimal decision making applications, which aims at retrieving points that are not dominated by other points in a data set. In this paper, we assume that **larger** values are preferred. Q^i denotes the i^{th} point and Q^i_k denotes the value on the k^{th} dimension of Q^i, then Q^i dominates Q^j, denoted as $Q^i \prec Q^j$, *iff* for each k, $Q^i_k \geq Q^j_k$ and for at least one k, $Q^i_k > Q^j_k$ $(1 \leq k \leq d)$. Fig. 1 shows a skyline example. The data set in Fig. 1 (left) consists of 5 points. Each point has two dimensions. We can see that $Q^4(4,4) \prec Q^3(4,2)$ as an example of dominance relationship between points. As shown in Fig. 1 (right), the skyline contains Q^1, Q^2 and Q^4.

Though skyline computation is particularly useful in multi-criteria decision making applications, it is inadequate to answer queries that need to analyze not only *individual* points but also their combinations [3, 4, 2, 6, 9]. Specifically, in many real-world applications, we need to find groups of points that are not dominated by other groups of equal size.

It is shown in [3, 4, 2, 6, 9] that a skyline group may consist of both skyline points and non-skyline points, all points in the data set have a chance to form a skyline group. Therefore, there are total C^k_n combinations, which are far more than the n candidates in traditional skyline computation. Moreover, the output size of skyline groups is far more than

points	A_1	A_2
Q^1	6	0
Q^2	0	6
Q^3	4	2
Q^4	4	4
Q^5	0	5

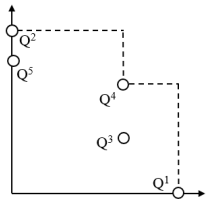

Figure 1: A skyline example

the size of skyline. The experimental results proposed in [6, 9] show that the output size is million scale even when the input is a few thousand points. The large output size is less informative and it may be hard for users to make a good, quick selection.

Motivation. The large output size promotes us to design an algorithm to select the best k skyline groups. Such k skyline groups should be most representative. Inspired by the top-k skyline queries, we quantify the concept of "representative" by counting the number of points dominated by the group. Here, we define that a point Q is dominated by a group G *iff* there exists at least one point Q' in G satisfying $Q' \prec Q$. We briefly summarize our contributions as follows:

- We propose a novel problem, top-k skyline groups query, so that the k skyline groups with maximal number of dominated points can be produced to facilitate user queries.
- We propose an efficient algorithm for processing k-SGQ, using several pruning techniques.
- We conduct extensive experiments to validate the effectiveness and efficiency of our proposals.

2. PRELIMINARY

In this section, we introduce the problem definition and related works.

2.1 Problem Definition

First, we introduce the definitions of dominance relationship between groups defined in [3, 4, 2, 6, 9]. We use $\prec_g$ to denote the dominance relationship between groups. Let $G \prec_g G'$ denote G dominates G'. The dominance relationship between groups defined in existing works can be divided into two kinds.

Definition 1. ($\prec_g$) [6] Assuming that $G = \{Q^1, Q^2, ..., Q^l\}$ and $G' = \{Q'^1, Q'^2, ..., Q'^l\}$ are two different groups with l

points. We say that $G \prec_g G'$, *iff* there exist two permutations of the l points for G and G', $G = \{Q^{u1}, Q^{u2}, ..., Q^{ul}\}$ and $G' = \{Q'^{u1}, Q'^{u2}, ..., Q'^{ul}\}$ satisfying that for each i, $Q^{ui} \preceq Q'^{ui}$ and for at least one i, $Q^{ui} \prec Q'^{ui}$ $(1 \leq i \leq l)$.

For instance in the Fig.1, since $Q^2 \prec Q^5$ and $Q^4 \prec Q^3$, thus $\{Q^2, Q^4\} \prec_g \{Q^3, Q^5\}$.

Definition 2. $(\prec_g)$ [3, 4, 2, 9] For an aggregate function f and a group $G = \{Q^1, Q^2, ..., Q^l\}$, then G is represented by a point Q, where $Q_j = f(Q_j^1, Q_j^2, ..., Q_j^l)$. For two distinct groups G and G', Q and Q' represents G and G' respectively. We define $G \prec_g G'$ *iff* $Q \prec Q'$.

In this paper, we study two kinds of aggregate function. The first one is strictly monotone, which means $f(Q_j^1, Q_j^2, ..., Q_j^l) > f(Q_j^{1'}, Q_j^{2'}, ..., Q_j^{l'})$ if $Q_j^i \geq Q_j^{i'}$ for every $i \in [1, l]$ and $\exists k$ such that $Q_j^k > Q_j^{k'}$, where $1 \leq k \leq l$. For the strictly monotone function, we study SUM in this paper. We also investigate aggregate functions that are not strictly monotone such as MAX and MIN. Fig. 2 shows the dominance relations under different aggregate functions.

	Points			SUM	MAX	MIN
G	Q^2(0,6)	Q^3(4,2)	Q^4(4,4)	(8,12)	(4,6)	(0,2)
G'	Q^3(4,2)	Q^4(4,4)	Q^5(0,5)	(8,11)	(4,5)	(0,2)
Dominance Relation				$G \prec_g G'$	$G \prec_g G'$	$G = G'$

Figure 2: Dominance relations under different aggregate functions

Based on the Definition 1 or Definition 2, skyline group is defined as follows:

Definition 3. **(GSkyline)** The l-point GSkyline consists of groups with l points that are not dominated by any other groups of the same size.

We define the problem of top-k skyline groups query in the following. To facilitate the presentation, we define a function $score(G)$ that counts the number of the points dominated by group G. Then we have:

$$score(G) = |\{Q \in D - G | \exists Q' \in G \wedge Q' \prec Q\}|$$

For instance, if $G = \{Q^2, Q^4\}$, $score(G) = 2$.

Definition 4. **(k-SGQ)** Top-k skyline groups query retrieves the set $S_K \subseteq GSkyline$ of k skyline groups with highest score values. Then we have:

$$\forall G \in S_K, \forall G' \in (GSkyline - S_K) \rightarrow score(G) \geq score(G')$$

Obviously, k-SGQ can be applied to find top-k skyline groups based on both Definition 1 and Definition 2.

2.2 Related Work

The most related works with regard to the concept of skyline groups queries are [3, 4, 2, 6, 8, 9]. [3, 4, 2, 8, 9] investigate the skyline groups query based on Definition 2 and Liu et al. [6] investigate the problem based on Definition 1. However, the output sizes of both definitions are large, which is a potential limitation of skyline group operator. To solve this, we propose an efficient algorithm to select top-k skyline groups.

The most related works to our k-SGQ are [5] and [8]. [5] proposes a top-k representative skyline points query. It aims to compute a set of k skyline points such that the total number of points dominated by one of the k skyline points is maximized. Obviously, it is inherently different from our problem. Moreover, since a skyline group may consist of both skyline points and non-skyline points. Therefore, the techniques proposed in [5] are not applicable to our problem. [8] proposes an algorithm to find top-k combinatorial skyline. In their work, a combinatorial skyline is a skyline group based on Definition 2. They rank skyline groups based on a predefined preferred attribute order. It only reports groups whose aggregate values for a certain attribute are the highest. Obviously, the problem proposed in [8] is also inherently different from our problem. Moreover, the ranking method proposed in [8] is not applicable to Definition 1, because a group cannot be represented by a point based on this definition. Therefore, [8] is orthogonal to our problem.

To the best of our knowledge, we are the first to address the problem of finding top-k skyline groups that dominate the maximum number of points.

3. COMPUTING TOP-*K* GSKYLINE

The brute-force method to compute k-SGQ is to enumerate all skyline groups and count the number of points dominated by each group, then select the best k skyline groups. For each skyline group we need $O(l \times n)$ time complexity to count the points dominated by the group. Let $|SG|$ denote the size of skyline groups, then time complexity of selecting best k groups is $O(|SG| \times log\, k)$. Therefore, the overall time complexity is $O(l \times n \times |SG| \times log\, k)$. Obviously, the brute-force method incurs high computation overhead.

3.1 The *k*-SGQ Algorithm

Let *Skyline* denote the set of points in the skyline. *Skyline* is an accompanying result when computing *GSkyline* [6, 9]. We summarize frequently used notions in Table 1.

LEMMA 1. *For Definition 1 and strictly monotone aggregate functions under Definition 2, if $G \in GSkyline$ and $G = \{Q^1, Q^2, ..., Q^l\}$, then for each $Q^i \in G$, we have $Q^i \in Skyline$ or $\exists Q^j \in G$ and $Q^j \in Skyline \rightarrow Q^j \prec Q^i$.*

PROOF. We prove by contradiction. Assume that $Q^j \prec Q^i$ *and* $Q^j \in Skyline$, if $Q^j \notin G$, we can use Q^j to replace Q^i in G, the new group is denoted as G'.

CASE 1. *For the Definition 1, since $Q^j \prec Q^i$ and all the other points are the same, then $G' \prec_g G$ which contradicts $G \in GSkyline$.*

CASE 2. *For a strictly monotone aggregate function f under Definition 2, since $Q^j \prec Q^i$, we assume that $Q_t^j > Q_t^i$. Then we have $f(Q_t^1, ..., Q_t^i, ..., Q_t^l) < f(Q_t^1, ..., Q_t^j, ..., Q_t^l)$. On other dimensions, we have $f(Q_{t'}^1, ..., Q_{t'}^i, ..., Q_{t'}^l) \leq f(Q_{t'}^1, ..., Q_{t'}^j, ..., Q_{t'}^l)$. Therefore, $G' \prec_g G$, which contradicts $G \in GSkyline$.*

Therefore, for Definition 1 and strictly monotone aggregate functions under Definition 2, if $G \in GSkyline$, then for each $Q^i \in G$, we can get that $Q^i \in Skyline$ or $\exists Q^j \in G$ *and* $Q^j \in Skyline \rightarrow Q^j \prec Q^i$. $\square$

LEMMA 2. *For MAX and MIN under Definition 2, if $\exists Q^i \in G$ $(G \in GSkyline)$ and Q^i is dominated by at least*

Table 1: The Summary of Notations

Notation	Description
D	A d-demonical data set
d	Number of dimensions
n	Number of points in D
Q^i	The i^{th} point in D
Q^i_j	The value on the j^{th} dimension of Q^i
$\prec$	Preference/dominance relation
$Skyline$	The skyline of data set D
l	Size of a group
$score(G)$	Number of points dominated by G

k points, then either (1) $\exists Q^j \in G$ and $Q^j \prec Q^i$ or (2) it is safe to prune G from GSkyline.

PROOF. Obviously, it is possible to have a point $Q^j \in G$ *and* $Q^j \prec Q^i$. In this situation we have $score(G) = score(G \setminus \{Q^i\})$.

In the second situation, all points dominate Q^i are not in G. If $Q^j \prec Q^i$, we use Q^j to replace Q^i in G, the new group is denoted as G'. Since $Q^j \prec Q^i$ and all the other points are the same, then $MAX(G') \preceq MAX(G)$ and $MIN(G') \preceq MIN(G)$. As G is a skyline group under MAX and MIN, we have $MAX(G') = MAX(G)$ and $MIN(G') = MIN(G)$ which means that G' is also a skyline group under MAX and MIN.

Moreover, we have $socre(G') \geq score(G)$. Since Q^i is dominated by at least k points, we have at least k skyline groups whose *socres* are equal or greater than $score(G)$. Therefore, it is safe to prune G from $GSkyline$. □

Let $dom(Q)$ denote the set of points dominated by point Q, then $score(G) = |\bigcup_{Q \in G} dom(Q)|$. In order to compute $score(G)$ efficiently, we maintain a bit vector for each point in the group, then we can employ fast bit-wise operations for much more efficient score computation.

Definition 5. **([Q])** $[Q]$ denotes the bit vector of Q. $[Q]$ has the length of $|D|$ bits, with one bit corresponding to a point in D. If a point Q^j is dominated by Q then the j^{th} bit is set to 1. Otherwise, the bit is set to 0.

Based on Definition 5, $score(G)$ equals the number of "1" in $[Q^1]|[Q^2]|...|[Q^l]$ ($G = \{Q^1, Q^2, ..., Q^l\}$). For instance in Fig. 1, $[Q^2] = 00001$, $[Q^4] = 00100$. If $G = \{Q^2, Q^4\}$, $score(G)$ equals the number of "1" in $[Q^2]|[Q^4] = 00101$. Therefore, $score(G) = 2$.

Based on Lemma 1 and Lemma 2, we do not need to compute $[Q]$ for every point in the data set. We use $(k-1)$-skyband [7] to denote the set of points that are dominated by at most $k-1$ points in a data set.

LEMMA 3. *For Definition 1 and strictly monotone aggregate functions under Definition 2, based on Lemma 1, we know that if $Q \in G$ and $Q \notin Skyline$, then Q has zero contribution to $score(G)$. Thus $[Q]$ is modified as follows:*

$$[Q] = \begin{cases} [Q], Q \in Skyline \\ 0...0, \quad Others \end{cases}$$

For MAX and MIN, we modify $[Q]$ in the following. Because if $Q \notin (k-1)$-skyband then either Q has zero contribution to $score(G)$ or it is safe to prune a candidate group containing Q, thus all points outside of the $(k-1)$-skyband will not affect the result of top-k skyline groups query.

$$[Q] = \begin{cases} [Q], Q \in (k-1)\text{-}skyband \\ 0...0, \quad Others \end{cases}$$

Definition 6. **(MaxScore)** $MaxScore$ denotes the upper bound of *score*. $MaxScore(G) = \sum_{Q \in G} |dom(Q)|$.

$$|dom(Q)| = \begin{cases} 0, & [Q] = 0...0 \\ number\ of\ "1"\ in\ [Q], & [Q] \neq 0...0 \end{cases}$$

Obviously, $MaxScore(G) \geq score(G)$.

LEMMA 4. *Let S_C be a candidate set containing k skyline groups and τ be the smallest score for all groups in S_C. For a specified group $G' \in GSkyline$ with $MaxScore(G') \leq \tau$, it can be safely pruned away as it cannot be an actual answer group for k-SGQ.*

Based on the above discussion, we propose Algorithm 1 to compute k-SGQ. In Algorithm 1 we maintain a priority queue of k skyline groups. Line 3 sets a threshold used for pruning. Then we compute the bit vectors for points in the $Skyline$ or $(k-1)$-skyband based on the skyline group definition and aggregate functions. Line 9 utilizes Lemma 4 to prune candidate groups, Line 10 utilizes bit-wise operations to compute $score(G)$ for candidate groups.

Algorithm 1: *The k-SGQ algorithm*

```
Input  : GSkyline, k;
Output: the result set S_K of k-SGQ on GSkyline
1  begin
2      PQ ← ∅; /* PQ is a priority queue sorting groups in the
           ascending order of their scores                    */
3      τ ← −1; /* τ is a threshold used for pruning          */
4      if the aggregate function is strictly monotone or under
       Definition 1 then
5          Compute the bit vectors for points in the Skyline;
6      if the aggregate function is MAX or MIN then
7          Compute the bit vectors for points in the
           (k − 1)-skyband;
8      for each group G in GSkyline do
9          if MaxScore(G) > τ then
10             if score(G) > τ then
11                 PQ.push(G);
12             if |PQ| > k then
13                 PQ.pop();
14                 τ ← PQ.top().score;
15     return PQ;
```

3.2 Time Complexity Analysis

The time complexity of computing bit vectors for points in the $Skyline$ and $(k-1)$-skyband is $O(|Skyline| \times n)$ and $O(|(k-1)\text{-skyband}| \times n)$ respectively. For each group G in $GSkyline$, we need $l-1$ bit-wise operations to get $score(G)$. The time complexity of updating the priority queue is $O(log\ k)$. Therefore, the time complexity of Algorithm 1 under Definition 1 and strictly monotone functions under Definition 2 is $O(|Skyline| \times n + (l-1) \times |SG| \times log\ k)$. The time complexity of Algorithm 1 for MAX and MIN is $O(|(k-1)\text{-skyband}| \times n + (l-1) \times |SG| \times log\ k)$. Obviously, the time complexity of Algorithm 1 is far less than the time complexity of the brute-force method.

4. EXPERIMENTAL EVALUATION

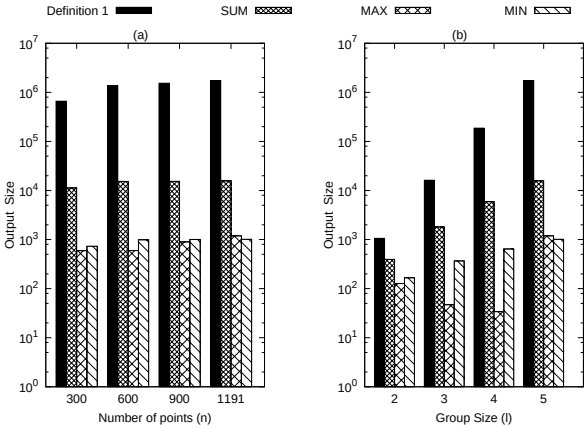

Figure 3: Output size of skyline groups based on Definition 1 and Definition 2 of varying n and l

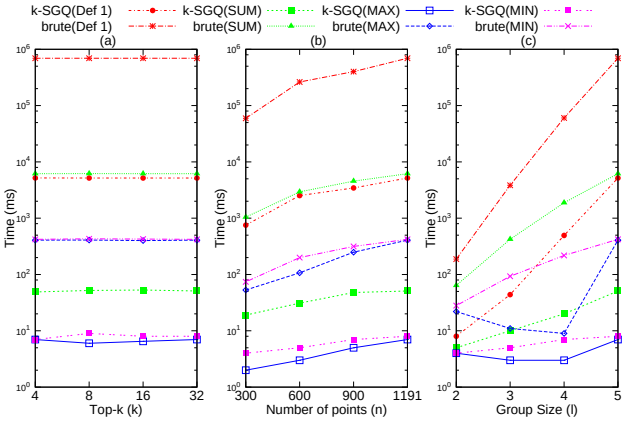

Figure 4: Performance of k-SGQ algorithm and brute-force method under different settings

In this section, we conduct extensive experiments to evaluate the run-time performance of Algorithm 1 under different settings. All our experiments are carried out on the same machine with 64GB memory and dual eight-core Intel Xeon E7-4820 processors clocked at 2.0Ghz. All algorithms are implemented in *C++*.

We continue to use the real data set adopted in [6]. The data set contains 1191 NBA players who are league leaders of playoffs. The data was extracted from http://stats.nba.com/leaders/alltime/?ls=iref:nba:gnav on 11/01/2016. Each player has five attributes that measure the player's performance. Those attributes are Points (PTS), Rebounds (REB), Assists (AST), Steals (STL) and Blocks (BLK).

We experiment with different settings, including number of best skyline groups k, number of points n and number of points in a group l. The settings of all these parameters are summarized in Table 2, where the default values are shown in bold. In every set of experiments, we only change one parameter, with the rest set to their defaults.

We compute skyline groups based on both Definition 1 and Definition 2. For Definition 2, we experiment with SUM, MAX and MIN. The output sizes of skyline groups are shown in Fig. 3. We can see that the output sizes based on both definitions are too large to make quick selections. Thus it is not trivial to design a top-k algorithm.

In Fig. 4 we present the performance of k-SGQ algorithm under different settings. We evaluate the performance of applying k-SGQ algorithm to find top-k skyline groups based on Definition 1 and Definition 2. From the three subfigures in Fig. 4 we can see that k-SGQ algorithm is efficient for both definitions under different settings. Therefore, our algorithm can be applied to all existing skyline group operators. Moreover, compared to the brute-force method, k-SGQ algorithm is much faster. For instance, there are 1720610 skyline groups generated from the NBA data set based on Definition 1, k-SGQ only needs 5 seconds to compute top-32 skyline groups while brute-force method needs 693 seconds. The experimental results show that for Definition 1 and SUM, k-SGQ algorithm is about average 134× and 121× speedup over the brute-force method respectively. For MAX and MIN, k-SGQ algorithm is about average 33× and 45× faster than the brute-force method respectively. Therefore, k-SGQ algorithm is efficient under different settings and can be applied to compute top-k skyline groups for all existing skyline group operators.

Table 2: Parameter Ranges and Default Values

Parameter	Range
k	4,8,16, **32**
n	300,600,900, **1191**
l	2,3,4, **5**

5. CONCLUSIONS

In this paper, we introduce a new and useful type of query, top-k skyline groups queries. The existing techniques cannot be applied to solve k-SGQ. We propose an efficient algorithm with several powerful pruning strategies. Moreover, we conduct extensive experiments to validate the efficiency our algorithm. Experimental results show that our algorithm reaches high performance under different settings.

6. ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grant No.61502511, No.61572514, and No.61502513.

7. REFERENCES

[1] S. Borzsony, D. Kossmann, and K. Stocker. The skyline operator. In *Proc. ICDE*, pages 421–430. IEEE, 2001.

[2] Y.-C. Chung, I.-F. Su, and C. Lee. Efficient computation of combinatorial skyline queries. *Information Systems*, 38(3):369–387, 2013.

[3] H. Im and S. Park. Group skyline computation. *Information Sciences*, 188:151–169, 2012.

[4] C. Li, N. Zhang, N. Hassan, S. Rajasekaran, and G. Das. On skyline groups. In *Proc. CIKM*, pages 2119–2123. ACM, 2012.

[5] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang. Selecting stars: The k most representative skyline operator. In *Proc. ICDE*, pages 86–95. IEEE, 2007.

[6] J. Liu, L. Xiong, J. Pei, J. Luo, and H. Zhang. Finding pareto optimal groups: group-based skyline. *Proc. VLDB*, 8(13):2086–2097, 2015.

[7] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive skyline computation in database systems. *TODS*, 30(1):41–82, 2005.

[8] I. F. Su, Y. C. Chung, and C. Lee. Top- k combinatorial skyline queries. In *Proc. DASFAA*, pages 79–93, 2010.

[9] N. Zhang, C. Li, N. Hassan, S. Rajasekaran, and G. Das. On skyline groups. *TKDE*, 26(4):942–956, 2014.