On Decisive Skyline Queries

Akrivi Vlachou[1], Christos Doulkeridis[2], João B. Rocha-Junior[3], and Kjetil Nørvåg[4]

[1] University of the Aegean, Greece
[2] University of Piraeus, Greece
[3] Universidade Estadual de Feira de Santana, Brazil
[4] Norwegian University of Science and Technology, Trondheim, Norway
avlachou@aegean.gr, cdoulk@unipi.gr, joao@uefs.br, noervaag@ntnu.no

Abstract. Skyline queries aim to identify a set of interesting objects that balance different user-specified criteria, i.e., that have values as good as possible in *all* specified criteria. However, objects whose values are good in only a subset of the given criteria are also included in the skyline set, even though they may take arbitrarily bad values in the remaining criteria. To alleviate this shortcoming, we study the decisive subspaces that express the semantics of skyline points and determine skyline membership. We propose a novel query, called *decisive skyline query*, which retrieves a set of points that balance all specified criteria. Our experimental study shows that the newly proposed query is more informative for the user.

Keywords: Skyline query · Decisive subspaces · Decisive skyline query.

1 Introduction

Skyline queries [2] constitute a powerful tool for data analysis and multi-objective optimization, as they enable balancing of different (and often conflicting) criteria specified by the user. Such queries return a set of data points (*skyline points*) that are not dominated by any other point in all dimensions. A point p *dominates* another point q, if p is better than or equal to q in all dimensions and strictly better than q in at least one dimension. Nevertheless, the skyline set contains also points that fail to balance among all given criteria, as we demonstrate in the following.

Example 1. (Motivating example) Assume that a tourist is interested in booking a hotel with a low price, a good ranking based on the customers' ratings, and nearby the beach. To this end, the tourist performs skyline analysis in an online hotel database in order to discover hotels that fulfill all criteria. For instance, a hotel called *City Center Hotel* may be included in the result due to its rank (best value), despite the fact that it may have the worst value in distance from the beach. Similarly, a hotel called *Sunset Hotel*, may have the worst rank among the skyline points and may be included in the result set because of its combined values in distance and price. However, had the user been really interested in

such a hotel, she would have specified as criteria only a subset of the dimensions, namely distance and price. In this paper, we argue in favor of a new query type, namely one that excludes from its results set hotels that may have arbitrarily bad values. We call this new query type *decisive skyline query*.

As shown in the example, the skyline query always returns the data point with the best value in one dimension, regardless of the values in the other dimensions, as this point cannot be dominated by any other point. Put differently, the skyline definition imposes "OR semantics" between the different criteria. In the hotel database, the skyline set contains the hotels that are the best trade-offs among (a) rank, price and distance, OR (b) rank, price, OR (c) price, distance, OR have (d) the minimum price, OR (e) the minimum rank, OR (f) the minimum distance. But this is not the objective of the user's search, since the user is looking for the best trade-offs among rank, price and distance.

An indirect consequence of the aforementioned "OR semantics" is that the skyline cardinality [5, 6, 16] increases rapidly with the dimensionality of the data space. The high cardinality of the skyline set originates from the fact that as the number of criteria increases, the combinations of different criteria increase exponentially. In turn, the probability that a point is dominated in all different combinations decreases, thus leading to more skyline points. Intuitively, it is more difficult to satisfy more criteria, therefore it would be expected that with increasing the number of criteria, the result size should decrease (or stay constant). Should we add too many criteria, none of the points will be able to satisfy all of them, thus resulting into an empty result set. In contrast to this intuition, the cardinality of skyline set increases with increasing dimensionality.

Most existing approaches focus on the effect of the problem and try to restrict the skyline cardinality. Towards this goal, different categories of approaches have been recently proposed, including (1) selecting k representative skyline points [8, 12, 13], (2) restricting the skyline cardinality by changing the dominance relationship [3], and (3) ranking the skyline points based on different metrics [4, 9, 14] or user-defined functions [1, 7].

In this paper, we take a radically different approach. We address what we consider to be the cause of the problem, and not the effect. To this end, we focus on the semantics of skyline queries (first studied by Pei et al. [11]). Informally, the *decisive subspaces* of a skyline point are responsible for the point being part of the skyline set, i.e., its values in these dimensions qualify it as skyline point. Capitalizing on this concept, we propose a novel query type, called *decisive skyline query*. We investigate two variants of the decisive skyline query, the *strict* variant, which returns only the subset of skyline points that have the full space as decisive subspace, and the *relaxed* variant, which returns also points with decisive subspaces that *cover* the entire data space. Interestingly, as a by-product, it turns out that the decisive skyline query does not suffer from increased output size for increased dimensionality. We emphasize that this is the first paper that focuses on the significance of retrieving points based on the properties of their decisive subspaces, since in [11] the aim was to find the subspace skyline points of all subspaces.

2 Problem Formulation

Given a data set P on a data space $\mathcal{D}$ defined by a set of m dimensions $\{d_1, \ldots, d_m\}$, a data object $p \in P$ is represented as an m-dimensional point p=$\{p[1], \ldots, p[m]\}$ where $p[i]$ is the value on dimension d_i. A point $p \in P$ dominates another point $q \in P$, denoted as $p \prec q$, if (1) on every dimension d_i, $p[i] \leq q[i]$; and (2) on at least one dimension d_j, $p[j] < q[j]$. The *skyline* $\mathcal{S}(P)$ is a set of points which are not dominated by any other point in P. Without loss of generality, we assume that skylines are computed with respect to min conditions on all dimensions and that all values are non-negative.

The notion of skyline can be extended to subspaces. Each non-empty subset U of $\mathcal{D}$ ($U \subseteq \mathcal{D}$) is referred to as a *subspace* of $\mathcal{D}$. The *skyline* of a subspace $U \subseteq \mathcal{D}$ is a set $\mathcal{S}_U(P) \subseteq P$ which are not dominated by any other point on subspace U. As shown in [11, 15], the skyline set of the full space does not contain all the subspace skyline points of the different subspaces. A skyline point q in $\mathcal{S}_U(P)$ is either a skyline point in $\mathcal{S}_V(P)$ (assuming $U \subset V$) or there exists another data point p, such that $p[i] = q[i]$ ($\forall d_i \in U$), that dominates q on the dimension set $V - U$.

2.1 Intuition of Decisive Subspaces

Let us first assume that the distinct value condition holds, which means that no two points share the same value in a given dimension (i.e., for any two points p and q of P it holds that $\forall d_i \in \mathcal{D} : p[i] \neq q[i]$). In this case, any subspace skyline point also belongs to the skyline set of the full space, which in turn simplifies the definition of the decisive skyline queries. Under the distinct value condition, the *decisive subspace* [11] of a skyline point p is defined as follows.

Definition 1. *(Decisive subspace) For a skyline point $p \in \mathcal{S}(P)$, a subspace U of $\mathcal{D}$ is called decisive, if (1) p is a subspace skyline in U ($p \in \mathcal{S}_U(P)$), and (2) there exists no subspace $V \subset U$ such that p is a subspace skyline point in V ($\nexists V \subset U$ such that $p \in \mathcal{S}_V(P)$).*

A skyline point p can have multiple decisive subspaces. We use $DecSub(p)$ to denote the set of decisive subspaces for a skyline point p. If a point p has a decisive subspace $U \subset \mathcal{D}$, then this fact alone promotes p to become a full space skyline, irrespective of p's values in dimension set $\mathcal{D} - U$. Obviously, such skyline points may not balance the remaining dimensions.

To address the problems of the semantics of the traditional skyline operator, we define the *strict decisive skyline* set $\mathcal{DS}(P)$ as the set of skyline points that have the full space $\mathcal{D}$ as their decisive subspace, i.e., $\mathcal{DS}(P) = \{p | p \in \mathcal{S}(P)$ and $DecSub(p) = \mathcal{D}\}$. Based on the definition of decisive subspaces for the case of distinct values, a skyline point p belongs to the decisive skyline set, if there does not exist any other subspace $U \subset \mathcal{D}$ for which p belongs to the subspace skyline set ($\nexists U \subset \mathcal{D}$ such that $p \in \mathcal{S}_U(P)$). We argue that decisive skyline points are guaranteed to have good values in all given criteria, in contrast to subspace skyline points.

The above definition imposes the semantics of decisive skyline sets in a strict (or rigid) way. A more relaxed variant, denoted $\widehat{\mathcal{DS}}(P)$, is also defined as follows: $\widehat{\mathcal{DS}}(P) = \{p | p \in \mathcal{S}(P)$ and $\bigcup_{(\forall U_i \in DecSub(p))} U_i = \mathcal{D}\}$. This *relaxed decisive skyline* set also includes points that belong to subspace skyline sets, as long as their decisive subspaces *cover* the full space. Thus, the relaxed decisive skyline points also balance all criteria, but possibly in different subspaces that cover the full space. Also, notice that by definition $\mathcal{DS}(P) \subseteq \widehat{\mathcal{DS}}(P)$.

	A	B	C
p_1	4	5	2
p_2	3	1	6
p_3	5	4	4
p_4	7	2	3
p_5	6	6	1
p_6	2	3	8
p_7	1	7	7

(a) Data set

	DecSub( )
p_1	{AC,AB}
p_2	{B,AC}
p_3	{ABC}
p_4	{AC}
p_5	{C}
p_6	{AB}
p_7	{A}

(b) *DecSub*

Fig. 1. Example of decisive subspaces.

Example 2. Consider a data space $\mathcal{D} = ABC$ and a data set P defined in $\mathcal{D}$ (Figure 1(a)). All points are skyline points and Figure 1(b) depicts their decisive subspaces. For p_7, subspace A is the decisive subspace, therefore the value of A is sufficient to qualify p_7 as a skyline point in the full space independently of its values in the other dimensions. Similar for p_2 and p_5 the decisive subspaces are B and C respectively. On the other hand, AC is also a decisive subspace for p_2, because p_2 appears in the subspace skyline of AC, and AC is not a super-set of B. Only point p_3 has the full space ABC as decisive subspace, and this is the only point in this example that belongs to the decisive skyline set $\mathcal{DS}(P) = \{p_3\}$. Points p_1 and p_2 may also be considered as good options, even though they do not belong to $\mathcal{DS}(P)$. For example, p_2 has the best value in dimension B, but also balances nicely dimensions AC, since it is in the subspace skyline in AC. Points p_1 and p_2, together with p_3, belong to the relaxed decisive skyline set $\widehat{\mathcal{DS}}(P) = \{p_1, p_2, p_3\}$.

2.2 Formal Definition of Decisive Subspaces

In the following, we withdraw the restriction on points taking distinct values. In the general case, the main difference is that there may exist subspace skylines that do not belong to the full space skyline points. Recall that a subspace skyline point $q \in \mathcal{S}_U(P)$ is either a skyline point in the full space or there exists another data point p, such that $p[i] = q[i]$ $(\forall d_i \in U)$, that dominates q on the dimension set $\mathcal{D} - U$. If such a point p exists, then the remaining $\mathcal{D} - U$ dimensions are important to determine whether q qualifies as a skyline point.

Definition 2. *(Maximal set of non-distinct points) Given a set of points G and set of dimensions U, we define as maximal set of non-distinct points the set:* $\mathcal{O}(G,U) = \{p_i | p_i \in P,\ \forall d_k \in U \text{ and } \forall p_j \in G : p_i[k] = p_j[k]\}$.

Based on the above definition, $\mathcal{O}(G,U)$ is the maximal set of points of P with identical values with the points of G in U, i.e., there exists no other point $q \in P$ with this property. The following definition is equivalent to the definition of [11].

Definition 3. *(Maximal skyline group) Given a set of points G and set of dimensions U, the pair $\{G,U\}$ is called maximal skyline group and is denoted as $SG(G,U)$, if it holds that (1) $\forall p_i \in G$ it holds that $p_i \in \mathcal{S}_U(P)$ (2) $\forall p_i, p_j \in G$ and $\forall d_k \in U$ $p_i[k] = p_j[k]$ (3) $\nexists d_k \in \mathcal{D} - U$ such that $\forall p_i, p_j \in G : p_i[k] = p_j[k]$ (4) $\nexists p_j \in P - G$ such that $\exists p_i \in G$ and $\forall d_k \in U : p_i[k] = p_j[k]$.*

Intuitively, $SG(G,U)$ is the maximal set of points with same values in U, these points are subspace skylines in U, and U is the maximal set of dimensions for which this set of points coincide.

In the following, we define the concept of decisive subspaces for a maximal skyline group.

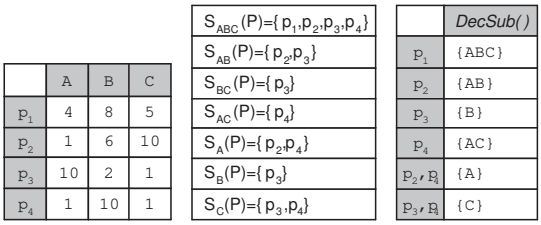
Definition 4. *(Decisive subspaces of maximal skyline group) Given a maximal skyline group $SG(G,U)$, a subspace $V \subseteq U$ is called decisive for $SG(G,U)$ if (1) $\forall p_i \in G$ it holds that $p_i \in \mathcal{S}_V(P)$ (2) $\mathcal{O}(G,V) = G$ (3) $\nexists V' \subset V$ such that conditions 1) and 2) hold for V'.*

2.3 Decisive Skyline Points

We denote the decisive subspaces of the maximal skyline group $SG(G,\mathcal{D})$ as $DecSub(G)$. A decisive subspace V of $SG(G,U)$ means that all points in G share the same values in U and are in the subspace skyline set for every subspace V' such that $V \subseteq V' \subseteq U$. We cannot conclude if all points of G belong to the skyline set of $\mathcal{D}$, since depending on the remaining dimensions some of them may be dominated. Only if they are incomparable in the remaining dimensions, then all of them will belong to the skyline set. Skyline points that belong to a maximal skyline group $SG(G,\mathcal{D})$ that has the full space as a decisive subspace are included to the skyline set based on the values of all given dimensions, regardless if these points form groups in some subspaces.

Definition 5. *(Strict decisive skyline points) A skyline point p belongs to the strict decisive skyline set $\mathcal{DS}(P) \subseteq \mathcal{S}(P)$ of a data set P, if there exists a maximal skyline group $SG(G,\mathcal{D})$ such that $p \in G$ and the decisive subspace of G is the full space ($DecSub(G) = \{\mathcal{D}\}$).*

Definition 6. *(Relaxed decisive skyline points) A skyline point p belongs to the relaxed decisive skyline set $\widehat{\mathcal{DS}}(P) \subseteq \mathcal{S}(P)$ of a data set P, if there exists a maximal skyline group $SG(G,\mathcal{D})$ such that $p \in G$ and the union of the decisive subspaces of G is the full space ($\bigcup_{\forall U_i \in DecSub(G)} U_i = \{\mathcal{D}\}$).*



(a) Data set (b) $S_U(P)$ (c) *DecSub*

Fig. 2. Example of decisive skyline set.

Example 3. Consider the data set P depicted in Figure 2(a). The decisive subspaces for each maximal skyline group $SG(G, U)$ are shown in Figure 2(c). We observe that point p_1 is the only point that belongs to the decisive skyline set. Point p_3 has B as decisive subspace. In turn, this means that p_3 belongs to the skyline set independently of its values in the other dimensions. In this example, p_3 has the worst value of all points in dimension A. On the other hand, point p_2 has AB as decisive subspace and not A, even though it is subspace skyline in A. This is because it coincides with p_4 in A and they form a group $\{p_2, p_4\}$ in that subspace. Still, the value of p_2 in dimension C does not influence whether p_2 belongs to the skyline set or not, thus this value can be arbitrarily high. Note that in this small example, the strict and relaxed decisive skyline points are the same.

3 Decisive Skyline Algorithm

A straightforward way to compute the decisive skyline set is to compute all maximum skyline groups and their decisive subspaces. Then, the points that belong to a group that have the full space as a decisive subspace can be easily determined. Computing all maximum skyline groups requires evaluating all $2^m - 1$ subspace skyline queries and requires multiple disk accesses on the same data. We refer to this approach as *Naive*.

As we will show in the following, we develop an algorithm for computing the strict decisive skyline set with two salient features. First, our algorithm avoids evaluating all subspace skyline queries, and instead evaluates only $m+1$ skyline queries. Second, assuming that data is indexed by a multidimensional index, we define an appropriate query that allows our algorithm to traverse the index at most once, retrieve a set of candidate points, and refine the result set in main-memory.

One important observation is that the strict decisive skyline points ($p \in \mathcal{DS}(P)$) are exactly those skyline points ($p \in \mathcal{S}(P)$) that for any $(m-1)$-dimensional subspace U they are either dominated ($p \notin \mathcal{S}_U(P)$) or share the same values in U with another data point p' ($p' =_U p$ and $p' \neq p$). We denote $p' =_U p$, if it holds that $\forall d_i \in U : p[i] = p'[i]$. However, in the simplest case

one skyline query and m subspace skyline queries need to be processed and the index must be accessed multiple times. To avoid this processing overhead, we identify a super-set of this set that can be efficiently retrieved by traversing the index structure at most once.

Definition 7. *(Enriched skyline) A point $p \in P$ is said to* partially dominate *another point $q \in P$ on $\mathcal{D}$, if (1) on every dimension $d_i \in \mathcal{D}$, $p[i] \leq q[i]$; and (2) on at least two dimensions $d_j, d_k \in \mathcal{D}$, $p[j] < q[j]$ and $p[k] < q[k]$. The* enriched skyline *of P is the set of points $e\mathcal{S}(P) \subseteq P$ which are not partially dominated by any other point.*

The above definition assumes that the enriched skyline is defined on a data space that contains at least two dimensions, i.e., $|\mathcal{D}| \geq 2$. An interesting observation is that the enriched skyline uses a slightly modified definition of dominance, that can be supported by any skyline algorithm with marginal overhead, by simply changing the function used for point dominance. We can prove that the enriched skyline set is sufficient to compute the strict decisive skyline set $\mathcal{DS}(P)$.

3.1 Algorithmic Description

We design an efficient algorithm, called *Decisive Skyline Algorithm* and denoted as *DSA*, for computing the strict decisive skyline $\mathcal{DS}(P)$ of a set of points P. The innovative features of our algorithm include that (a) DSA operates only on a subset of the data set P, namely the enriched skyline set, that is both easy to compute and guaranteed to include all decisive skyline points, and (b) DSA computes the decisive skyline by efficient processing of the underlying subspace skyline queries without the need to access the disk repeatedly. DSA extends the well-known branch-and-bound (BBS) algorithm for skyline queries [10].

The pseudocode describing DSA is shown in Algorithm 1. The algorithm takes as input a data set P indexed by an R-tree $\mathcal{R}$ and produces the decisive skyline set $\mathcal{DS}(P)$ as output. First, BBS is executed on the R-tree that indexes P, and it populates the main-memory R-tree $\mathcal{M}$ with the enriched skyline points (and only those). In addition, the skyline set $\mathcal{S}(P)$ is retrieved (line 2). Notice that the two parameters of the $BBS()$ call in the pseudocode correspond to the index used by BBS and the subspace which is processed respectively. Then, DSA executes m subspace skyline queries on the main-memory R-tree $\mathcal{M}$ iteratively (lines 3-20). For each $(m-1)$-dimensional subspace U, DSA exploits the progressive property of BBS and retrieves subspace skyline points sorted by their distance to the origin in subspace U (line 7) and places them in a buffer $\mathcal{B}$ (line 9). This guarantees that points with identical values in U will be processed in a batch. Processing of a batch of points includes examining each point p in $\mathcal{B}$ and checking whether it is a candidate point (lines 11-15). If so, then we test if there exists another point p' with identical values on the $m-1$ dimensions of U, but different on the last dimension. If such a point p' does not exist, the point p can safely be discarded from the candidate points (line 13). The same procedure is repeated until all subspace skyline points are processed or the candidate list ($\mathcal{S}(P)$) gets empty (line 6).

Algorithm 1 *Decisive Skyline Algorithm (DSA)*

input: The R-tree index $\mathcal{R}$ built on data set P
output: The decisive skyline set $\mathcal{DS}(P)$

```
1: M ← null, B ← ∅ //M:main-memory R-tree, B:buffer
2: (S(P), M) ← BBS(R, D)
3: for i = 1...m do
4:    U ← D − d_i //U:current subspace of m − 1 dimensions
5:    tmpDist ← −1
6:    while has next point BBS(M, U) and S(P) ≠ ∅ do
7:       q ← next point of BBS(M, U)
8:       if Dist_U(q) = tmpDist then
9:          B = B ∪ q
10:      else
11:         for all p ∈ B do
12:            if (p ∈ S(P)) and (∄p' ∈ B : ∀d_k ∈ U p[k] = p'[k] and p[i] ≠ p'[i]) then
13:               S(P) ← S(P) − p
14:            end if
15:         end for
16:         B = {q}
17:      end if
18:      tmpDist ← Dist_U(q)
19:   end while
20: end for
21: return S(P)
```

DSA exploits the main-memory R-tree internally used by BBS, in order to efficiently compute the decisive skyline set. Thus, DSA takes practically *for free* the index structure that is constructed by BBS during the skyline computation, thereby making the subsequent execution of subspace skyline queries extremely efficient. Moreover, as all decisive skyline points belong to the skyline set, we modify further BBS, so that only skyline points are reported as result ($\mathcal{S}(P)$), even though the main-memory R-tree indexes the enriched skyline points.

In practice, DSA processes m subspace skyline queries (of dimensionality $m-1$) using the main-memory R-tree, for excluding non-decisive skyline points from the already computed skyline set by BBS. Notice that the main-memory R-tree indexes only the enriched skyline points, therefore the execution of subspace skyline queries is more efficient compared to processing on the entire data set P on disk.

4 Experimental Evaluation

In this section, we provide an experimental study of the decisive skyline query. All algorithms are implemented in Java and the experiments run on a machine with 2x Intel Xeon X5650 Processors (2.66GHz), 128GB.

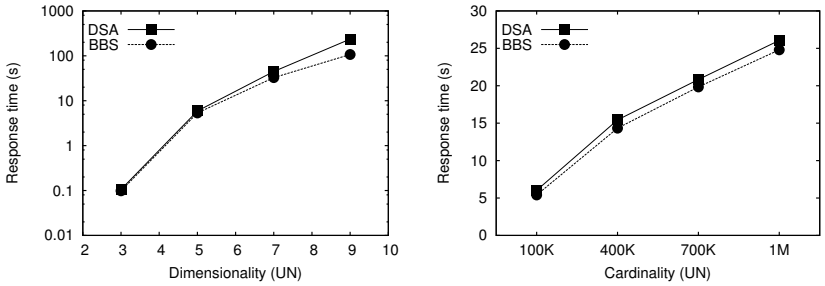
4.1 Qualitative Study

We perform skyline analysis on data extracted from DBLP, in order to discover researchers with significant number of publications on a combination of conferences. The data set contains data that reflect DBLP entries before 15/10/2008. We use the authors as points represented in a multidimensional space defined by the number of publications in selected conferences (dimensions). Major conferences from different research areas are selected as criteria that need to be balanced. We underline the strict decisive skyline points, while relaxed decisive skyline points are shown using bold. Each researcher is represented as a 3d point with values equal to the number of publications for each of the selected conferences, and higher values are preferable.

Id	Name	SIGMOD	PODS	CIKM	*DecSub*()
1	**Divyakant Agrawal**	**14**	**7**	**11**	{S,P,C}
2	Jeffrey F. Naughton	29	9	1	{S,P}
3	Amr El Abbadi	7	8	12	{P,C}
4	Jiawei Han	26	0	8	{S,C}
5	Dan Suciu	14	15	2	{P,C}
6	Serge Abiteboul	15	24	0	{S,P}
7	Michael J. Carey	36	3	0	{S}
8	Jeffrey D. Ullman	17	16	0	{S,P}
9	**Divesh Srivastava**	**28**	**10**	**3**	{S,C}, {P,C}
10	Yehoshua Sagiv	8	29	1	{P}
11	**Christos Faloutsos**	**19**	**4**	**8**	{S,P,C}
12	Raghu Ramakrishnan	28	14	1	{S,P}
13	Surajit Chaudhuri	33	8	0	{S,P}
14	Philip S. Yu	18	1	18	{C}
15	David J. DeWitt	33	1	1	{S,C}

Table 1. $\widehat{\mathcal{DS}}(P)$: in bold, $\mathcal{DS}(P)$: underlined.

Table 1 shows the skyline set for {SIGMOD,PODS,CIKM}. *Divyakant Agrawal* and *Christos Faloutsos* are the strict decisive skyline points and they balance nicely all criteria, compared to the remaining skyline points. By inspecting the result set, we observe that several researchers do not truly balance all given criteria (dimensions). Instead, they may balance subsets of the dimensions only, but not all of them. For instance, *Jeffrey D. Ullman* nicely balances SIGMOD and PODS, but not CIKM. The same holds for both *Serge Abiteboul* and *Raghu Ramakrishnan*, who are also experts in data management. On the other hand, *Yehoshua Sagiv* is included in the result due to the extremely high number of PODS publications. The decisive skyline query manages to exclude these points from the result set, thus returning only points that truly balance all criteria. It is likely that if a user were interested in only a subset of the criteria, she would have posed a 2d query with only the criteria of interest instead. On the other hand,



(a) Varying dimensionality (b) Varying cardinality

Fig. 3. Comparative study: Performance of DSA versus BBS.

Divesh Srivastava will be included in the relaxed decisive skyline set, because he is a subspace skyline in subspaces {PODS,CIKM} and {SIGMOD,CIKM} that cover the full space, thus he manages to balance all given criteria.

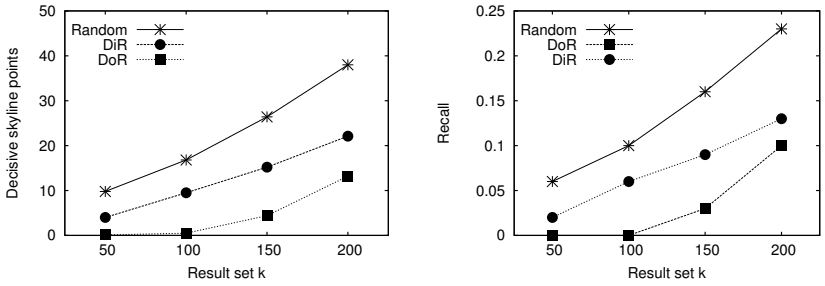
4.2 Performance of Decisive Skyline Algorithm

In the following, we study the cost of computing the strict decisive skyline query (using DSA), compared with the cost of computing the skyline query (using the BBS algorithm). Although in this latter experiment DSA and BBS compute two different result sets, the experiment aims to answer the following interesting question: *how much is the overhead of computing decisive skyline points, compared to the computation of the skyline set?*

In Figure 3(a), we study the effect of increased dimensionality, using a synthetic data set of 1M records following a uniform data distribution. We report the average results over 10 different instances of the data set. When increasing the dimensionality, the difference in response time between DSA and BBS is small and increases slowly. In Figure 3(b), we vary the cardinality from 100K to 1M, and we observe that the difference in time between DSA and BBS is small and constant, which is the expected result since the major impact in DSA is the number of subspace computations that is fixed in this setting. In summary, our finding is that DSA retrieves the strict decisive skyline set with a slightly increased cost compared to a state-of-the-art skyline algorithm (BBS [10]) that retrieves the traditional skyline set.

4.3 Comparison with Representative Skylines

Thereafter, we try to answer the following research question: *can the set of decisive skyline points be obtained by existing algorithms proposed for representative skyline computation?* To this end, we compare our algorithm against two well-known skyline representative algorithms, namely dominance-based representative (DoR) [8] and distance-based representative (DiR) [13]. Both approaches



(a) Number of decisive skyline points (b) Recall wrt. decisive skyline points

Fig. 4. Comparison with algorithms for representative skylines.

take as a input the number k of skyline points that are selected as representatives. In addition, we use a *Random* algorithm that selects k representative skyline points from the skyline set at random. Figure 4(a) shows the number of strict decisive skyline points retrieved by the representative skyline algorithms as the value of k increases. In this experiment, the size of strict decisive skyline set is 192 and the skyline cardinality is 952. As shown in the chart, both DoR and DiR fail to retrieve the decisive skyline points. In fact, even a random selection of skyline points (Random) retrieves more decisive skyline points than DoR and DiR. This demonstrates that the existing skyline representative algorithms do not take into account the semantics of the decisive skyline points and select completely different skyline points compared to our approach. Figure 4(b) shows the recall that each representative skyline algorithm achieves, when using the strict decisive skyline points as correct result. As depicted in the chart, the recall of the representative skyline algorithms is very low, which demonstrates that these algorithms do not try (not even implicitly) to identify decisive skyline points. In fact, even a random selection of skyline points (Random) retrieves more decisive skyline points than DoR and DiR. This demonstrates that the existing skyline representative algorithms select completely different skyline points compared to our approach.

5 Conclusions

In this paper, we exploit the semantics of skyline points and propose the *decisive skyline query*. Capitalizing on the concept of decisive subspaces, we define two variants of the decisive skyline set that are subsets of the skyline set. Points belong to the decisive skyline set due to their values in *all* user-specified criteria and provide interesting trade-offs. As a positive by-product and in contrast to skyline cardinality, the cardinality of the decisive skyline query is not significantly affected by dimensionality, thus leading to smaller result sets even for high-dimensional data. Our evaluation demonstrates the performance of our algorithm and that the decisive skyline query returns interesting points to the user.

References

1. Bartolini, I., Ciaccia, P., Oria, V., Özsu, M.T.: Flexible integration of multimedia sub-queries with qualitative preferences. Multimedia Tools Appl. **33**(3), 275–300 (2007)
2. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: Proc. of ICDE. pp. 421–430 (2001)
3. Chan, C.Y., Jagadish, H.V., Tan, K.L., Tung, A.K.H., Zhang, Z.: Finding k-dominant skylines in high dimensional space. In: Proc. of SIGMOD. pp. 503–514 (2006)
4. Chan, C.Y., Jagadish, H.V., Tan, K.L., Tung, A.K.H., Zhang, Z.: On high dimensional skylines. In: Proc. of EDBT. pp. 478–495 (2006)
5. Chaudhuri, S., Dalvi, N.N., Kaushik, R.: Robust cardinality and cost estimation for skyline operator. In: Proc. of ICDE. p. 64 (2006)
6. Godfrey, P.: Skyline cardinality for relational processing. In: Proc. of FoIKS. pp. 78–97 (2004)
7. Lee, J., won You, G., won Hwang, S.: Personalized top-k skyline queries in high-dimensional space. Information Systems **34**(1), 45–61 (2009)
8. Lin, X., Yuan, Y., Zhang, Q., Zhang, Y.: Selecting stars: the k most representative skyline operator. In: Proc. of ICDE (2007)
9. Lu, H., Jensen, C.S., Zhang, Z.: Flexible and efficient resolution of skyline query size constraints. IEEE TKDE (2011)
10. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive skyline computation in database systems. ACM TODS **30**(1), 41–82 (2005)
11. Pei, J., Jin, W., Ester, M., Tao, Y.: Catching the best views of skyline: A semantic approach based on decisive subspaces. In: Proc. of VLDB. pp. 253–264 (2005)
12. Sarma, A.D., Lall, A., Nanongkai, D., Lipton, R.J., Xu, J.J.: Representative skylines using threshold-based preference distributions. In: Proc. of ICDE. pp. 387–398 (2011)
13. Tao, Y., Ding, L., Lin, X., Pei, J.: Distance-based representative skyline. In: Proc. of ICDE. pp. 892–903 (2009)
14. Vlachou, A., Vazirgiannis, M.: Ranking the sky: Discovering the importance of skyline points through subspace dominance relationships. DKE **69**(9), 943–964 (2010)
15. Yuan, Y., Lin, X., Liu, Q., Wang, W., Yu, J.X., Zhang, Q.: Efficient computation of the skyline cube. In: Proc. of VLDB. pp. 241–252 (2005)
16. Zhang, Z., Yang, Y., Cai, R., Papadias, D., Tung, A.: Kernel-based skyline cardinality estimation. In: Proc. of SIGMOD. pp. 509–522 (2009)