Optimal Preference Elicitation for Skyline Queries over Categorical Domains

Jongwuk Lee[1], Gae-won You[1], Seung-won Hwang[1], Joachim Selke[2], and Wolf-Tilo Balke[2]

[1] Department of Computer Science and Engineering, POSTECH, Korea
[2] L3S Research Center, Leibniz Universität Hannover, Germany
{julee,gwyou,swhwang}@postech.edu
{selke,balke}@L3S.de

Abstract. When issuing user-specific queries, users often have a vaguely defined information need. Skyline queries identify the most "interesting" objects for users' *incomplete preferences*, which provides users with intuitive query formulation mechanism. However, the applicability of this intuitive query paradigm suffers from a severe drawback. Incomplete preferences on domain values can often lead to impractical skyline result sizes. In particular, this challenge is more critical over categorical domains. This paper addresses this challenge by developing an iterative elicitation framework. While user preferences are collected at each iteration, the framework aims to both minimize user interaction and maximize skyline reduction. The framework allows to identify a reasonably small and focused skyline set, while keeping the query formulation still intuitive for users. All that is needed is answering a few well-chosen questions. We perform extensive experiments to validate the benefits of our strategy and prove that a few questions are enough to acquire a desired manageable skyline set.

1 Introduction

The information need of users in today's databases and information systems has evolved from SQL-style exact match queries to answering vague queries. To address this need, new query paradigms like top-k retrieval or skyline queries have been recently studied. These paradigms assess the grades of match in all database objects with respect to a given query, and only identify the best matching results.

More specifically, the strengths of two paradigms are complementary. Top-k retrieval returns only the best k objects based on a user-specific *utility function* combining scores with respect to all queried attributes. While top-k queries always provide a focused and manageable set, it is difficult for end-users to define an exact utility function for their individual preferences. In contrast, skyline queries do not require users to define a utility function, and simply identify "interesting" objects that are not "dominated" by any other objects. While this intuitive query formulation has been a key strength of skyline queries, it is impossible for users to control the size of skyline. In particular, when the number of

Table 1. Toy dataset for Example 1

ID	*type*	*color*	*brand*
o_1	convertible	red	Ferrari
o_2	sedan	red	Ferrari
o_3	convertible	blue	Ferrari
o_4	sedan	blue	Toyota
o_5	roadster	blue	Honda

queried attributes increases, the size of skyline also increases exponentially, *i.e.*, *curse of dimensionality.* This challenge is especially more critical over categorical domains.

This paper deals with skyline queries over *categorical domains* in which the challenge of skyline queries is more critical. Although both paradigms have been mostly applied for numerical domains in the previous literatures (*e.g.*, minimizing *price* or *distance*), these can also be used for categorical domains as well (*e.g.*, maximizing a preference on favorite *color* or *brand*). To illustrate, Example 1 describes how skyline queries work in categorical domains.

Example 1. Consider a customer shopping for an ideal car with respect to three attributes *type*, *color* and *brand.* Suppose that a user gives specific preferences that he/she prefers 'convertible' to 'sedan' for *type*, 'red' to 'blue' for *color*, and 'Ferrari' to 'Honda' for *brand.* Based on these preferences, we identify car o_1 as one of the best choices, *i.e.*, a skyline object, from the toy dataset in Table 1. This means o_1 is superior to o_2 and o_3 in all dimensions, *i.e.*, o_1 *dominates* o_2 and o_3. However, the user preferences are not sufficient to determine a preference between o_1 and o_4, or o_1 and o_5, *i.e.*, o_1 is *incomparable* with o_4 or o_5.

As Example 1 illustrates, in practical scenarios, the amount of preference information available to query processing is usually limited, because specifying all relationships requires considerable effort for the user. Missing relationships are thus interpreted as *indifference*, or equal importance for the user. As a result, skyline query results will typically include all the incomparable objects, due to incomplete user preferences.

This paper studies the problem of eliciting preferences enough to acquire a concise skyline result set. In particular, we use the cardinality of different domain values with respect to the database instance (and *a priori* knowledge on user preferences, if exists). This makes users elicit more useful preferences with minimal user efforts. Ideally, such an elicitation process achieves both minimizing user interaction and maximizing skyline reduction. We thus aim at developing and evaluating an optimal elicitation process. In summary, this paper has the following contributions:

- We study preference elicitation in numerical and categorical domains and design an optimal elicitation strategy (Section 2)

- We develop Framework *MaxPrune* to identify skylines with reasonable size by implementing our optimal elicitation strategy. (Section 3)
- We validate effectiveness and efficiency of Framework *MaxPrune*. (Section 4)

This paper is organized as follows. Section 2 presents preliminaries on qualitative preference and elicitation model over categorical domains. Section 3 proposes a framework adopting optimal elicitation method in the given problem setting, and Section 4 validates Framework *MaxPrune*. Section 5 briefly reviews existing efforts related to our work. Finally, Section 6 discusses our future work.

2 Preliminaries

This section states preliminaries to help understand our framework. Let D be a data space with finite n attributes $\{D_1, D_2, \ldots, D_n\}$, where D_i denotes a set of possible *domain values* on i^{th} attribute. Specifically, let D be *possible alternatives*, *i.e.*, $D := D_1 \times D_2 \times \ldots \times D_n$, and A be *actual alternatives* as a subset of D, *i.e.*, $A \subseteq D$. An alternative $a = (a_1, \ldots, a_n)$ is contained in a product set $A := A_1 \times A_2 \times \ldots \times A_n$. A *weak order* is denoted as $\succeq$ on the set of alternatives A, by setting $a \succeq b$ if and only if a is equal to or more preferred than b. The asymmetric part and symmetric part of weak order, denoted as $\succ$ and $\sim$, correspond to *strict order* and *indifference*, respectively.

2.1 Qualitative Preferences

We first discuss strengthes and weaknesses of qualitative preferences. Specifically, given alternatives a and b, it clearly requires much less cognitive effort to tell which one among $a \succ b$, $b \succ a$, and $a \sim b$ holds. This ignores any numerical values and solely considers an induced weak order. However, for large D and A, it seems hopeless to ask the user about his/her preferences in a qualitative way, since there are $\binom{|D|}{2}$ pairs to be compared. An exception is a numerical attribute domains with an inherent order based on which users can express preference straightforwardly, *e.g.*, ascending order of "price".

For practical aspects, we introduce *ceteris paribus* semantics which provides an intuitive meaning [24,13]. For instance, saying "red $\succ_{\text{color}}$ blue" means "The user prefers red cars to blue ones, if everything else is equal". Since this is exactly the meaning of just saying "red cars are better than blue ones", stating preferences in terms of attribute value comparisons is highly intuitive. Based on *ceteris paribus* semantics, preference monotonicity between alternatives can be constructed over multi-attribute domains. This construction rule exploits *Pareto aggregation*, a relational operator that maps a sequence of weak orders into a binary relation on a set. Specifically, let $W_1, \ldots, W_k$ be weak orders on set $S := S_1 \times \ldots \times S_k$. The operator of Pareto aggregation, denoted as Par, is defined as follows: For $a = (a_1, \ldots, a_k), b = (b_1, \ldots, b_k) \in S$, it is $(a, b) \in \text{Par}(W_1, \ldots, W_k)$ if and only if $a_i W_i b_i$ is true, for any $1 \leq i \leq k$. It is easy to show that $\text{Par}(W_1, \ldots, W_k)$ is derived from weak orders on S.

Returning to the problem of reconstructing a total order $\succeq$ on A from attribute orders $\succeq_1, \dots, \succeq_n$, it is known that $\mathrm{Par}(\succeq_1, \dots, \succeq_n)$ is the best reconstruction of $\succeq$. We have the following reasons: First, $\mathrm{Par}(\succeq_1, \dots, \succeq_n)$ is always a subset of $\succeq$. Second, for any superset of $\mathrm{Par}(\succeq_1, \dots, \succeq_n)$, there exists a utility function inducting weak orders $\succeq_1, \dots, \succeq_n$. We thus will base our model on $\succeq_1, \dots, \succeq_n$ and $\mathrm{Par}(\succeq_1, \dots, \succeq_n)$. For the sake of representation, we simplify $\mathrm{Par}(\succeq_1, \dots, \succeq_n)$ into $\succeq_{\mathrm{Par}}$. Also, The symmetric part and asymmetric part of $\succeq_{\mathrm{Par}}$ correspond to $\succ_{\mathrm{Par}}$ and $\sim_{\mathrm{Par}}$, respectively.

The final questions to be answered are then: What are the "best" alternatives? What alternatives should be returned by the database system when the attribute orders $\succeq_i$ are known? To answer these questions, we adopt skyline queries leveraging Pareto aggregation, and define the "best" actual alternatives to be exactly those that are not strictly dominated in A with respect to $\mathrm{Par}(\succeq_1, \dots, \succeq_n)$. More formally, we define dominance and skyline, respectively. (These definitions are consistent with the definition of skyline used in all the existing skyline work.)

Definition 1. *An alternative $a \in A$ strictly dominates an alternative $b \in A$ if and only if $a_i \succeq_i b_i$, for any index i, and there is an index j such that $a_j \succ_j b_j$.*

Definition 2. *An alternative $a \in A$ is a skyline object if and only if there is no alternative $b \in A$ that strictly dominates a.*

2.2 Preference Elicitation

The term *preference elicitation* refers to the task of collecting information about the user's preferences. In the existing skyline work, it is usually assumed that $\succeq_1, \dots, \succeq_n$ are complete total orders for preference elicitation. However, this assumption is unrealistic over categorical domains.

We first discuss how to model preference elicitation for collecting more "informative" user preferences. In particular, we model preference elicitation as an *iterative* process in which the user answers which one among $a \succ_i b$, $b \succ_i a$, and $a \sim_i b$ holds, where $a, b \in D_i$. User preferences are consistently collected for t iterations, which is essentially binary relations $\succeq_1^{(t)}, \dots, \succeq_n^{(t)}$, where index $t \in \mathbb{N}$ refers to the time index of elicitation iteration. As preference elicitation accumulates monotonically, preference knowledge also accumulates in any elicitation step, *i.e.*, for any i and t, the relation $\succeq_i^{(t+1)}$ is a superset of $\succeq_i^{(t)}$. Since we know that the "true" orders $\succeq_i$ are reflexive and transitive, an elicited order $\succeq_i^{(t)}$ also must have these properties. We formally state an elicitation step at time t. (For simplicity, we denote the existing derived notations as follows: $\succ_i^{(t)}$, $\sim_i^{(t)}$, $\succeq_{\mathrm{Par}}^{(t)}$, $\succ_{\mathrm{Par}}^{(t)}$, and $\sim_{\mathrm{Par}}^{(t)}$.)

Definition 3. *Given weak orders $\succeq_1^{(t)}, \dots, \succeq_n^{(t)}$, an elicitation step from time t to time $t+1$ is the following procedure:*

(1) Choose attribute values $a, b \in D_i$ on i^{th} attribute, where neither $a \succeq_i^{(t)} b$ nor $b \succeq_i^{(t)} a$ is true.
(2) Ask the user which one among $a \succ_i b$, $b \succ_i a$, and $a \sim_i b$ is true.

*(3) If $a \succ_i b$ is true, define $\succeq_i^{(t+1)}$ to be the transitive closure of $\succeq_i^{(t)} \cup \{(a,b)\}$.
If $b \succ_i a$ is true, define $\succeq_i^{(t+1)}$ to be the transitive closure of $\succeq_i^{(t)} \cup \{(b,a)\}$.
If $a \sim_i b$ is true, define $\succeq_i^{(t+1)}$ to be the transitive closure of $\succeq_i^{(t)} \cup \{(a,b),(b,a)\}$.*

The elicitation process starts at time $t = 0$ with weak orders $\succeq_i^{(0)}$, which can contain initial information on user's preferences. That is, it can contain domain-specific preferences shared by all users, or personalized preference information based on a user profile.

2.3 Optimal Elicitation Method

The hardest part of preference elicitation is asking the user the right questions. Some questions may result in a large decrease of skyline size when stepping from $\mathrm{SKY}\big(A, \succeq_{\mathrm{Par}}^{(t)}\big)$ to $\mathrm{SKY}\big(A, \succeq_{\mathrm{Par}}^{(t+1)}\big)$, while other questions might not. For example, we know nothing about the user's preferences, but we know A to contain roughly as many blue cars as red cars. It thus would be a reasonable strategy to ask the first question about the preference relationship between attribute values "red" and "blue". If the user is not indifferent between both, the answer to this question can be expected to result in a large decrease of skyline size (assuming a good-natured data distribution in A). Based on this property, we formally state an elicitation method as follows:

Definition 4. *An elicitation method E is a deterministic algorithm that takes initial attribute weak orders $\succeq_1^{(0)}, \ldots, \succeq_n^{(0)}$ as input and performs a sequence of elicitation steps until time t is reached with $\succeq_1^{(t)}, \ldots, \succeq_n^{(t)}$ being weak orders.*

Clearly, the optimality of elicitation method depends on the distribution of actual alternatives A and prior knowledge of typical user preferences. To represent these notions, we introduce the following notations: Let $\mathcal{W}_i$ be the set of all possible weak orders on D_i. Also, let $\mathcal{W} := \mathcal{W}_1 \times \cdots \times \mathcal{W}_n$, and let $\mathcal{Q}$ be a probability distribution on $\mathcal{W}$, where $Q = (Q_1, \ldots, Q_n)$ denotes a random variable having distribution $\mathcal{Q}$. We use distribution $\mathcal{Q}$ to model prior knowledge of user preferences. (We will later discuss different elicitation decisions based on the distribution $\mathcal{Q}$ in details.) Up to now, we assumed both the elicitation method E and the user's preferences $w \in \mathcal{W}$ used for answering the elicitation questions to be fixed. To allow more precision in further definitions, we extend notations by making these assumptions explicit. We thus denotes notation $\succeq_{\mathrm{Par}}^{(t)}$ at time t as $\succeq_{E,w,\mathrm{Par}}^{(t)}$

Our goal finds a minimal sequence of questions with the smallest number t for $\Big|\mathrm{SKY}\big(A, \succeq_{E,w,\mathrm{Par}}\big)\Big| \leq k$ by $\mathrm{steps}_{E,w}(k)$, where k means user-specific retrieval size. This setting is similar to that of top-k retrieval except for two major differences: First, while a result set in top-k retrieval is assumed to contain the "best" k objects, in our setting a set of all the optimal objects (of size at most k) is required to be returned. Second, in top-k retrieval the user has to be proactive and state her/his preferences in advance. In our case, the active part is played by an internal framework. We formally state the optimality of elicitation method.

Definition 5. *Denote the class of all elicitation methods that take an additional* $\mathbb{N}$*-valued input argument by* $\mathcal{C}$*. For any* $E \in \mathcal{C}$*, write* $E(k)$ *to denote the behavior of* E *when given input* k*. An elicitation method* $E \in \mathcal{C}$ *is optimal (with respect to* D*,* A*, and* $\mathcal{Q}$*) if and only if*

$$\mathbb{E}\Big(\text{steps}_{E(k),Q}(k)\Big) \leq \mathbb{E}\Big(\text{steps}_{F(k),Q}(k)\Big)$$

holds, for $k \in \mathbb{N}$*,* $\forall F \in \mathcal{C}$*, and the number of questions* $\mathbb{E}(\cdot)$*.*

The idea underlying this definition is that an optimal elicitation method should ask questions in a way such that the number of expected questions needed to reach the target skyline size is as small as possible. Note that the definition of optimality is relative to D, A, and $\mathcal{Q}$. In practice, we are looking for a general optimal algorithm that work well regardless of the choice of D, A, and $\mathcal{Q}$. This corresponds to a greedy algorithmic approach, where in any elicitation step, the most desirable question leading to the best possible reduction in skyline size will be chosen. Without loss of generality, the next definition presents a step-optimal elicitation method, which also guarantees global optimality. (We will discuss this property in Section 3.)

Definition 6. *A greedy elicitation method* E *is called step-optimal (with respect to* D*,* A*, and* $\mathcal{Q}$*) if and only if, when given attribute preorders* $\succeq_1^{(t)}, \ldots, \succeq_n^{(t)}$ *for input, the algorithm* E *maximizes the term*

$$\Big|\text{SKY}\big(A, \succeq_{Par}^{(t)}\big)\Big| - \mathbb{E}\bigg(\Big|\text{SKY}\big(A, \succeq_{F,Q',Par}^{(t+1)}\big)\Big|\bigg)$$

over all greedy elicitation methods F *(given the same input), with* $Q' = (Q'_1, \ldots, Q'_n)$ *as a random variable that is distributed according to* $\mathcal{Q}$ *conditioned on the fact that* Q'_i *is a superset of* $\succeq_i^{(t)}$*, for any* i*.*

3 Optimal Elicitation Framework

In this section, we implement an optimal elicitation framework (discussed in Section 2.3) in a restricted problem setting. In particular, our framework aims at maximizing *expected pruning cardinality* (Definition 5) at each elicitation step, based on which the greedy elicitation strategy has global optimality.

3.1 Problem Setting

At each iteration, our framework shows a *sample pair*[1] (a, b) such that $a, b \in D_i$, and prune out objects that are never qualified as skyline objects. This means, when a user selects a over b , every object o with non-preferred attribute value

[1] Note the sample corresponds to a question on some pair a and b, asking which among $a \succ b$, $b \succ a$, or $a \sim b$ holds.

b can be pruned, if there exists another object o' having the same values as o in all dimensions except for $o'(D_i) = a$. We formally state this pruning process as follows. (Due to the space limitation, we leave all the proofs to our technical report [20].)

Lemma 1 (Pruning Process). *For user preference $a \succeq_i b$ on $a, b \in D_i$, we safely prune out object o such that $o(D_i) = b$, if there exists object o' such that $o'(D_i) = a$ and $\forall j (1 \leq j \leq n, j \neq i) : o(D_j) = o'(D_j)$.*

Note that, we assume that there always exists such dominating object o', which simplifies the pruning process. We argue that this assumption is often true in real-life data, as highly preferred values often have high frequency as well (as Zipf's law similarly observed, *i.e.*, the frequency of any word is roughly inversely proportional to its rank in the number of frequency). This observation implies that a dominating (or highly preferred/ranking) object o' is highly likely to exist as in our assumption.

3.2 Framework *MaxPrune*

We first derive a sample (a, b) maximizing *pruning cardinality* $\mathrm{PC}(\cdot)$ at the t^{th} step. Specifically, pruning cardinality $\mathrm{PC}(\cdot)$ means the number of pruned objects by Lemma 1. Let s_i denote a sample at the i^{th} step. Note that $\mathrm{PC}(\cdot)$ is conditional for prior elicitation– For a set of skyline objects after an elicitation of the t^{th} step, denoted as $\mathrm{SKY}(A, \succeq^{(t)}_{s_t,w,\mathrm{Par}})$, $\mathrm{PC}(s_t)$ of sample $s_t = (a, b)$ depends on prior samples, $s_1, \ldots, s_{t-1}$. In particular, when a user answers his/her preference on s_t, $w = a \succeq^{(t)}_i b$, we remove objects with value b from a set of current skyline objects $\mathrm{SKY}(A, \succeq^{(t-1)}_{\mathrm{Par}})$. $\mathrm{PC}(s_t, w | \, \mathrm{SKY}(A, \succeq^{(t-1)}_{\mathrm{Par}}))$ is thus denoted as

$$\left|\mathrm{SKY}\big(A, \succeq^{(t-1)}_{\mathrm{Par}}\big)\right| - \left|\mathrm{SKY}\big(A, \succeq^{(t)}_{s_t,w,\mathrm{Par}}\big)\right|.$$

Observe that $\mathrm{PC}(s_t, w | \, \mathrm{SKY}(A, \succeq^{(t-1)}_{\mathrm{Par}}))$ is maximized when the number of objects in $\mathrm{SKY}(A, \succeq^{(t-1)}_{\mathrm{Par}})$ with less preferred attribute value is maximal. We formally state this property as follows:

Lemma 2 (Maximizing Pruning Cardinality). *For user preference $w = a \succeq^{(t)}_i b$ of $s_t = (a, b)$ on $a, b \in D_i$, $\mathrm{PC}(s_t, w | \, \mathrm{SKY}(A, \succeq^{(t-1)}_{\mathrm{Par}}))$ is maximized, when the number of objects with less preferred value b is maximal.*

Based on Lemma 2, we discuss how to decide a sample maximizing pruning cardinality. In fact, since pruning cardinality depends on user preference w, we develop a *probabilistic* framework with the following two scenarios, with and without *a-priori* knowledge based on distribution $\mathcal{Q}$ for user preference.

– **Without a-priori knowledge on $\mathcal{Q}$:** With no a-priori knowledge, we assume that a probability that each value in a sample pair is selected is equal chances, *i.e.*, $\frac{1}{2}$. That is, this probability implies that the user equally prefers either one among the two values. Distribution $\mathcal{Q}$ for user preferences thus follows uniform distribution on $\mathcal{W}$. For instance, when showing a sample ('Ferrari', 'Honda') for '*brand*', we assume that a user prefers each value with $\frac{1}{2}$ probability.

– **With a-priori knowledge on $\mathcal{Q}$:** On the other hand, we may have *a-priori* knowledge on user preferences, *e.g.*, such as *query frequency* from prior query logs. Based on this we can model user preferences more realistically. The distribution $\mathcal{Q}$ for user preferences shows different distributions according to $\mathcal{W}$. For instance, when a relative preference probability between 'Ferrari' and 'Honda' is p_f and p_h respectively, the probability of choosing 'Ferrari' over 'Honda' can be computed as $p_f/(p_f + p_h)$, while that of choosing 'Honda' is $p_h/(p_f + p_h)$[2].

For ease of understanding, we first develop our framework assuming *no a-priori knowledge*, which is later extended to consider also *a-priori knowledge.* In particular, we develop the notion of *expected pruning cardinality* based on the probabilistic assumption that the selected probability of the two values is equivalent.

Theorem 1 (Expected Pruning Cardinality). *Assuming no a priori knowledge, expected pruning cardinality* $\mathrm{EPC}(s_t | \mathrm{SKY}(A, \succeq_{\mathrm{Par}}^{(t-1)}))$ *is maximized, when presenting sample* $s_t = (a, b)$ *in which the number of objects with value* a *and* b *in* $\mathrm{SKY}(A, \succeq_{\mathrm{Par}}^{(t-1)})$ *is maximal.*

Theorem 1 can be straightforwardly extended to consider the case of a-priori knowledge. Let the relative preferences of a and b in $s_t = (a, b)$ be p_a and p_b, respectively. In that case, expected pruning cardinality $\mathrm{EPC}(s_t | \mathrm{SKY}(A, \succeq_{\mathrm{Par}}^{(t-1)}))$ is maximized, when choosing sample $s_t = (a, b)$ which maximizes $p_b \times c_a$ and $p_a \times c_b$ in which c_a and c_b is the cardinality of objects with value a and b in $\mathrm{SKY}(A, \succeq_{\mathrm{Par}}^{(t-1)})$.

Based on Theorem 1, we derive the global optimality of greedy elicitation at each step. Specifically, when choosing a sample with the highest expected pruning cardinality at each step, global pruning cardinality also guarantees optimality. To prove this property, we first show properties on the order of samples selected from each step in Lemma 3 and Lemma 4. Based on these properties, we can show that global pruning optimality in Theorem 2 can be derived from selecting maximal expected pruning cardinality at each step.

Lemma 3 (Exchange of adjacent samples). *Given a sequence of samples* $S = (s_1, ..., s_t)$*, changing the order of an arbitrary pair of adjacent samples has no effect on the sum of the expected pruning cardinality.*

Lemma 4 (Ordering Independence). *Changing the order of a given sequence of samples* $S = (s_1, ..., s_t)$ *has no effect on the sum of the expected pruning cardinality.*

[2] Assume that this relative preference is independent regardless of other relative preferences.

Table 2. Illustration of *MaxPrune*

type	*card.*	*color*	*card.*	*brand*	*card.*
convertible	40	red	60	Ferrari	35
sedan	30	blue	40	Honda	35
roadster	20	-	-	Toyota	30
sports car	10	-	-	-	-

Theorem 2 (Global Pruning Optimality). *Choosing* s_i $(1 \leq i \leq t)$ *maximizing expected pruning cardinality at each iteration leads to optimal sampling* $S = \{s_1, ..., s_t\}$*, which maximizes overall expected pruning cardinality* $\mathrm{SUM}(S) = \left|\mathrm{SKY}\big(A, \succeq^{(0)}_{\mathrm{Par}}\big)\right| - \left|\mathrm{SKY}\big(A, \succeq^{(t)}_{\mathrm{Par}}\big)\right|$.

We now develop our framework based on Theorem 2. We name our framework as *MaxPrune*, where overall expected pruning cardinality is maximized by identifying an optimal sample at each steep. We briefly describe how Framework *MaxPrune* works. As an initial state, all current skyline objects are initialized as the entire set of data instances. Framework *MaxPrune* then follows the following three steps– First, it selects a sample with the highest expected pruning cardinality from all current skyline objects. Second, it collects a user preference with respect to the given sample, based on which it prunes all dominated objects having a non-preferred value from the current skyline. Lastly, it updates the cardinalities in each dimension. The processes are repeated until the number of skyline objects is reduced to at most k.

To illustrate, we describe how Framework *MaxPrune* works over our example dataset in Table 2. First, we consider samples with the highest expected pruning cardinality, *e.g.*, 'convertible' and 'sedan' for *type*, 'red' and 'blue' for *color*, and 'Ferrari' and 'Honda' for *brand*. Among these, we decide to obtain a preference elicitation on 'red' and 'blue' first, since its expected pruning cardinality (Theorem 1) is the highest, *e.g.*, $\frac{1}{2}(60 + 40)$. Once the elicitation result is obtained, for instance 'red' $\succ_{brand}$ 'blue', for each object with 'red', we can prune out objects with 'blue' sharing the same remaining attribute values. For the remaining objects, we then update the cardinality of attribute values for each attribute. Framework *MaxPrune* continues this iterative process until the size of skylines is reduced to k or less. We formally state Framework *MaxPrune* as follows:

1. Initialize $\mathrm{SKY}(A, \succeq^{(0)}_{\mathrm{Par}})$ as the entire data set.
2. Select the most effective sample as a pair of values with the highest expected pruning cardinality (Theorem 1).
3. Elicit preference information on the sample, and according to the user preference, prune out "dominated" objects from current skylines. (Lemma 1)
4. Update the cardinality of attribute values for each attributes.
5. Repeat step 2, 3, and 4 until $|\,\mathrm{SKY}(A, \succeq^{(t)}_{\mathrm{Par}})| \leq k$.

Table 3. Parameters for Experimental Setup

Parameter	Value : Default
Database Size N	$100K$
Dimensionality n	[3,7] : 4
Number of distinct values m	[3,7] : 4
Retrieval Size $k(\%)$	[1,20] : 5
Skewness of Data z	[0,2] : 1
Skewness of Query Frequency z'	[0,2] : 1
Kendall τ distance d	[0,1] : 0.5

4 Experimental Evaluation

This section validates the effectiveness and efficiency of frameworks *MaxPrune* and *MaxPruneQF*[3] using various synthetic datasets. Our experiments were carried out on a Intel(R) Xeon(TM) machine with 3.20 GHz dual processors and 1GB RAM running LINUX. Our algorithms were implemented in C++ language.

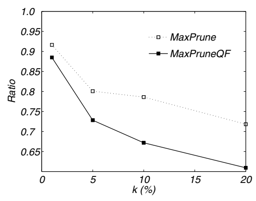
4.1 Data and Preference Generation

For the purpose of extensive evaluations, we generate synthetic datasets by varying experiment settings, including the data size N, the number of distinct attribute values m, and the user-specified retrieval size $k\%$ (of N), as described in Table 3. Especially, we randomly generate m distinct attribute values and query frequency for each dimension, according to Zipfian and Uniform distributions, varying the skewness from $z = 0$ (uniform) to $z = 2$. Note that we generate datasets to follow the assumption described in Section 3.1, *i.e.*, there exists at least one object for every attribute value combination. Specifically, we first populate one object for every alternative, and then generate N objects according to Zipfian distribution.

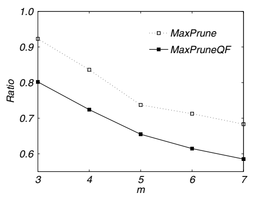
We then generate user preferences and interactions to compare our frameworks *MaxPrune* and *MaxPruneQF* with and without a priori knowledge, *e.g.*, query frequency $\mathcal{Q}$. In particular, we randomly generate query frequency for each dimension based on Zipfian distribution with the skewness $z' = [0, 2]$. We then follow user interactions on $\mathcal{Q}$, to prefer values in the descending order of query frequency for each dimension. Note, if this descending order of query frequency coincides with the descending order of cardinality, *i.e.*, when two orders are perfectly correlated, the behavior of *MaxPrune* and *MaxPruneQF* will be identical. We thus observe their behavior over the varying correlations. In particular, we adopt the Kendall τ distance $d = \frac{1}{n}\sum_1^n \mathcal{K}_i(\mathcal{W}_i, \mathcal{O}_i)$ [15], a widely-adopted metrics to quantify the correlation between two orderings $\mathcal{W}_i$ and $\mathcal{O}_i$. $\mathcal{K}_i(\mathcal{W}_i, \mathcal{O}_i)$ is defined as follows:

$$\mathcal{K}_i(\mathcal{W}_i, \mathcal{O}_i) = \frac{\sum_{(p,q)} |\mathcal{W}_i(p) > \mathcal{W}_i(q) \wedge \mathcal{O}_i(p) < \mathcal{O}_i(q)|}{m(m-1)/2}$$

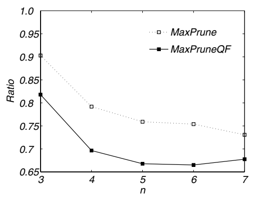
[3] Framework *MaxPruneQF* extends *MaxPrune* to use prior knowledge, *i.e.*, realistic distribution of user preferences, as discussed in Section 3.



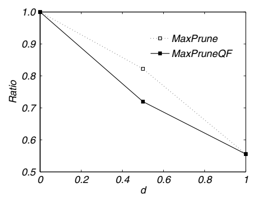
(a) over $k(\%)$



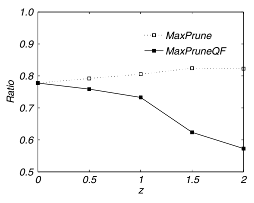
(b) over m



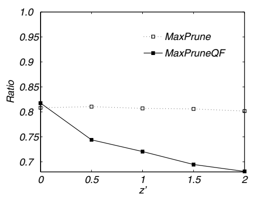
(c) over n



(d) over d



(e) over z



(f) over z'

Fig. 1. Number of iterations over varying parameters

where (p, q) denotes a possible pair of values and $\mathcal{W}_i(p)$ and $\mathcal{O}_i(p)$ represent each position in the respective ordering.

4.2 Experimental Results

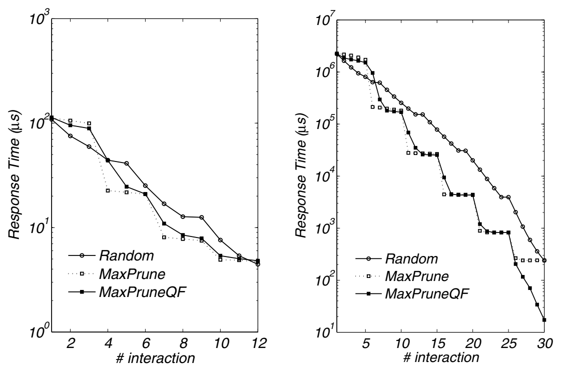
In this section, we report our experiment results validating the pruning effectiveness and efficiency of our proposed frameworks. Table 3 describes experiment settings used. In particular, we adopt the following performance metrics:

- **Effectiveness:** We use the number of iterations until we identify the k best results, averaged over 100 runs. We report relative performance against that of Framework *Random* which randomly selects a sample to present.
- **Efficiency:** We measure the runtime performance of our framework at each iteration, compared with that of Framework *Random*.

Pruning Effectiveness: Figure 1 reports the effectiveness of our frameworks over varying parameters. Note, the $y-$axis represents the relative number of iterations of our proposed frameworks, compared to that of *Random*.

$$Ratio = \frac{\#\text{ iterations with our framework}}{\#\text{ iterations with } \textit{Random}}.$$

First, Figure 1(a) reports our results over varying retrieval size $k\%$. Observe that, Framework *MaxPrune* and *MaxPruneQF*, by minimizing user interactions, significantly outperforms Framework *Random*– For instance, when $k = 20\%$, Framework *MaxPrune* saves 28% and 38% from Framework *Random* and Framework *MaxPruneQF* respectively. Our frameworks similarly dominate the baseline



(a) over $m = 4$, $n = 4$ (b) over $m = 6$, $n = 6$

Fig. 2. Number of remaining skyline tuples

approach over m and n, which can be observed in Figure 1(b) and (c) respectively. Also, observe that our frameworks scale more gracefully compared to *Random*. That is, the performance gaps increase as m and n increase– For instance, when $m = 7$ in Figure 1(b), *MaxPrune* and *MaxPruneQF* save about 27% and 32% from *Random* respectively, while they save 10% and 18% when $m = 3$. Similarly, when $n = 7$ in Figure 1(c), this saving reaches up to 32% and 41% respectively.

Second, Figure 1(d) validates the effectiveness over varying correlation distance (quantified as a Kendall τ distance d discussed above). As already analyzed above, *MaxPrune* and *MaxPruneQF* behave identically when $d = 1$, which can be observed from Figure 1(d). As d increases, the performance gap increases, which reaches up to 55% when $d = 0.5$. Third, Figure 1(e) and (f) validate the effectiveness with respect to the skewness of datasets and query frequency, respectively. Observe from the figures that, the relative effectiveness of our proposed frameworks increase as the skewness increases, especially for *MaxPruneQF*. For instance, when $z = 2$ and $z' = 2$ Framework *MaxPruneQF* saves 43% and 32% from that of *Random*, respectively. In summary, we validate that our framework significantly outperforms *Random* in datasets with high cardinality and dimensionality, especially in the presence of high skewness in both data and query frequency.

Runtime Performance: Figure 2(a) validates the efficiency of our frameworks by reporting an average response time for each iteration over the default setting (Table 3). Note that y-axis is log-scaled. As the figure reports, the response time of our frameworks is comparable to *Random* at all iterations, which is impressive considering *Random* blindly picks a random sample. Further, our framework starts to outperform *Random*, as the number of remaining skyline objects rapidly decreases over the iterations. For instance, after the 4^{th} iteration, our framework begins to deal with a much smaller sample pool and outperforms *Random* from this point on. Similarly, Figure 2 reports results for extended parameters with $m = 6$ and $n = 6$.

5 Related Work

We summarize related work on skyline computation and the representation of user preferences.

Skyline computation: Skyline queries have been first studied as maximal vectors in [19]. Later, Börzsönyi at el. [5] introduced skylines queries in database applications. Next, Tan et al. [23] proposed progressive skyline computation using auxiliary structures. Kossmann et al. [18] improves (D&C) algorithm, and proposed nearest neighbor (NN) algorithm. Similarly, Papadias et al. [22] developed branch and bound skyline (BBS) algorithm which achieves I/O optimal property. Meanwhile Chomicki et al. [10] developed sort-filter-skyline (SFS) algorithm leveraging pre-sorting lists, and Godfrey et al. [12] proposed linear elimination-sort for skyline (LESS) algorithm with attractive average-case asymptotic complexity. Recently, there have been active research efforts to address "curse of dimensionality" problem of skyline queries [6,7,21] using inherent properties of skylines such as *skyline frequency*, *k-dominant skylines*, and *k-representative skylines.* All these efforts, however, focused only on numerical domains with inherent orders, and did not consider skyline queries over categorical domains.

Preference foundation: For representing a variety of user preferences, Kießling [16,17] proposed a framework using binary preference relations. Similarly, Chomicki [9,10] developed a preference model, which consists of a basic preference *winnow* operator and its combinators. These preference models refer that *qualitative* models are more "intuitive" than *quantitative* models [11,14], which is consistent with our view. Meanwhile, Balke et al. [3,4,1,2] studied how to use *incomplete* preference information for skyline queries: In particular, [3,4] studied how to identify skylines over user-specified partial orders. More recently, [1] extended the notion of equivalence to include the inter-attribute equivalence, and [2] discussed a sophisticated user interface in the cooperative process of identifying partial orders. Meanwhile, Chen and Pu [8] summarizes methods eliciting user preferences. However, this framework does not address how to collect and leverage user-specific preferences. Our work helps users to elicit the most informative partial information on their preferences.

6 Future Work

We plan to extend our work in several ways. First, we can extend our framework into general environment combining numerical and categorical domains. Our technical report [20] discusses this extension in more details. Second, we want to develop new pruning heuristics that are less restricted than the one used in *MaxPrune* yet computationally feasible, to support sparse data sets. Lastly, we plan to explore how elicitation methods can make use of more complicated a-priori knowledge on the preference distribution (*e.g.*, dependencies in probability between attributes or attribute values).

Acknowledgments

This research has been partially supported by the Korean Research Foundation Granted funded by the Korean Government (MOEHRD, Basic Promotion Fund; KRF-2007-331-D00377) and the German Research Foundation (DFG) within the Emmy Noether Programme.

References

1. Balke, W.-T., Güntzer, U., Lofi, C.: Eliciting matters– controlling skyline sizes by incremental integration of user preferences. In: Kotagiri, R., Radha Krishna, P., Mohania, M., Nantajeewarawat, E. (eds.) DASFAA 2007. LNCS, vol. 4443, pp. 551–562. Springer, Heidelberg (2007)
2. Balke, W.-T., Güntzer, U., Lofi, C.: User interaction support for incremental refinement of preference-based queries. In: RCIS (2007)
3. Balke, W.-T., Güntzer, U., Siberski, W.: Exploiting indifference for customization of partial order skylines. In: IDEAS (2006)
4. Balke, W.-T., Güntzer, U., Siberski, W.: Getting prime cuts from skylines over partially ordered domains. In: BTW (2007)
5. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: ICDE (2001)
6. Chan, C.-Y., Jagadish, H., Tan, K., Tung, A.K., Zhang, Z.: On high dimensional skylines. In: Ioannidis, Y., Scholl, M.H., Schmidt, J.W., Matthes, F., Hatzopoulos, M., Böhm, K., Kemper, A., Grust, T., Böhm, C. (eds.) EDBT 2006. LNCS, vol. 3896, pp. 478–495. Springer, Heidelberg (2006)
7. Chan, C.-Y., Jagadish, H., Tan, K.-L., Tung, A.K., Zhang, Z.: Finding k-dominant skyline in high dimensional space. In: SIGMOD (2006)
8. Chen, L., Pu, P.: Survey of preference elicitation methods. In: EPFL Technical Report (2004)
9. Chomicki, J.: Querying with intrinsic preferences. In: Jensen, C.S., Jeffery, K.G., Pokorný, J., Šaltenis, S., Bertino, E., Böhm, K., Jarke, M. (eds.) EDBT 2002. LNCS, vol. 2287, Springer, Heidelberg (2002)
10. Chomicki, J., Godfery, P., Gryz, J., Liang, D.: Skyline with presorting. In: ICDE (2003)
11. Fishburn, P.C.: Utility Theory for Decision Making. Wiley, Chichester (1976)
12. Godfrey, P., Shipley, R., Gryz, J.: Maximal vector computation in large data sets. In: VLDB (2005)
13. Hansson, S.O.: What is ceteris paribus preference? Journal of Philosophical Logic 25(3), 307–332 (1996)
14. Keeney, R.L., Raiffa, H.: Decisions with Multiple Objectives. Preferences and Value Tradeoffs. Wiley, Chichester (1976)
15. Kendall, M.: A new measure of rank correlation. In: Biometrica (1938)
16. Kießling, W.: Foundations of preferences in database systems. In: VLDB, pp. 311–322 (2002)
17. Kießling, W., Köstler, G.: Preference SQL- design, implementation, experience. In: VLDB, pp. 311–322 (2002)
18. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: VLDB (2002)
19. Kung, H.T., Luccio, F., Preparata, F.: On finding the maxima of a set of vectors. Journal of the Association for Computing Machinery (1975)

20. Lee, J., You, G., Hwang, S., Selke, J., Balke, W.-T.: Optimal preference elicitation for skyline queries over categorical domains. POSTECH Technoical Report (2008), `http://ids.postech.ac.kr/~parfum/skyline.pdf`
21. Lin, X., Yuan, Y., Zhang, Q., Zhang, Y.: Selecting stars: The k most representative skyline operator. In: ICDE (2007)
22. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progessive algorithm for skyline queries. In: SIGMOD (2003)
23. Tan, K., Eng, P., Ooi, B.C.: Efficient progressive skyline computation. In: VLDB (2001)
24. von Wright, G.H.: The Logic of Preference. An Essay. Edinburgh University Press (1963)