The Skyline of a Probabilistic Relation

Ilaria Bartolini, Paolo Ciaccia, Marco Patella
DEIS - Università di Bologna, Italy
E-mail: {i.bartolini, paolo.ciaccia, marco.patella}@unibo.it

Technical Report:
DEIS-MMDBG-11-10

Abstract
In a deterministic relation $R$, a tuple $u$ dominates tuple $v$ if $u$ is no worse than $v$ on all the attributes of interest, and strictly better than $v$ on at least one attribute. This notion of Pareto domination is at the heart of skyline queries, that return the set of undominated tuples in $R$. Unlike previous approaches, in which the skyline of a probabilistic relation is not univocally defined, being it dependent on a threshold parameter, in this paper we demonstrate that, given a semantics for linearly ranking probabilistic tuples, the concept of skyline is well-defined even in the probabilistic case. Our approach exploits the order-theoretic definition of Pareto domination and preserves the three fundamental properties the skyline has in the deterministic case: 1) it equals the union of all top-1 results of monotone scoring functions, 2) it requires no additional parameter, and 3) it is insensitive to actual attribute scales. We then show how domination among probabilistic tuples (or $P$-domination for short) can be efficiently checked by means of a set of rules. We detail such rules for all most notable semantics for ranking of probabilistic tuples. Since computing the skyline of a probabilistic relation is a time-consuming task, we introduce a family of algorithms for checking $P$-domination rules in an optimized way. Our experiments show that these algorithms can dramatically reduce the actual execution times with respect to a naïve evaluation, which makes skyline queries applicable also to large probabilistic datasets.

1 Introduction
Uncertain data management has recently become a very active area of research, due to the huge number of relevant applications it has. Among them,
data extraction from the Web [7] (due the imprecision of the extraction tools),
data integration [14] (where data in different sources are matched with a certain
degree of confidence), moving objects [10], sensor networks [9], etc.

According to a commonly adopted model, uncertain data can be repre-
sented through probabilistic relations, in which each tuple has also a proba-
bility to appear [1, 21, 2]. A probabilistic relation can be viewed as a compact
representations of a set of possible worlds, each of them being a deterministic
relation formed by a subset of the original tuples. In order to capture cor-
relation among tuples, a set of generation rules is also present, e.g., to state
that some tuples are mutually exclusive, which limits the number and the
structure of the possible worlds.

In recent years, several works have focused on extending different query
types to probabilistic databases [21, 13]. In this paper we concentrate on skyline
queries [6], whose relevance in supporting multi-criteria decision analysis
is well known [8, 5]. The skyline of a relation $R$ is the set of undominated
(or Pareto-optimal) tuples in $R$, where tuple $u$ dominates tuple $v$ if $u$ is no
worse than $v$ on all the attributes of interest, and strictly better than $v$ on at
least one attribute. The appeal of skyline comes from the observation that it
consists of all and only top-1 results obtainable from scoring functions that
are monotone in the skyline attributes, thus providing users with an overall
picture of what are the best alternatives in a relation.

As an example, consider a traffic-monitoring application that collects data
by means of a set of radars and stores them in a database, a sample of last-
hour recording being shown in Figure 1.1 Because of radar locations, a
same car cannot be detected by two distinct radars within an interval of one
hour. This implies, for instance, that tuples $t_2$ and $t_3$ are mutually exclusive.
Each radar reading has associated a Prob value, representing the overall
confidence one has in the reading. A skyline query on the Time and Speed
attributes would find those readings that, at the same time, are the most
recent ones and concern high-speed cars. In the deterministic case it would be $\text{Sky}(R) = \{t_1, t_2, t_3, t_5, t_{11}\}$, since all other tuples are dominated (e.g.,
t_1 dominates t_9). This implies, for instance, that there exists a monotone
scoring function on the Time and Speed attributes that is maximized by $t_1$,
thus making $t_1$ the top-1 result for such function, but this is not the case for
$t_9$ (i.e., $t_9$ can never be a top-1 tuple).

In the probabilistic case, defining what the skyline could be is indeed
challenging. Intuitively, one should consider not only the ranking attributes
(i.e., Time and Speed), but also the confidence one has on each reading. This
interplay between probability and ranking attributes is a well-known issue

\footnote{This example was also considered by previous works on top-k queries [22, 17].}
<table>
<thead>
<tr>
<th>TID</th>
<th>Plate No</th>
<th>Radar Loc</th>
<th>Time</th>
<th>Speed</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₁</td>
<td>X-123</td>
<td>L1</td>
<td>10:53</td>
<td>90</td>
<td>0.2</td>
</tr>
<tr>
<td>t₂</td>
<td>W-246</td>
<td>L2</td>
<td>10:50</td>
<td>100</td>
<td>0.15</td>
</tr>
<tr>
<td>t₃</td>
<td>W-246</td>
<td>L3</td>
<td>10:40</td>
<td>95</td>
<td>0.1</td>
</tr>
<tr>
<td>t₄</td>
<td>Z-456</td>
<td>L1</td>
<td>10:32</td>
<td>110</td>
<td>0.1</td>
</tr>
<tr>
<td>t₅</td>
<td>Z-456</td>
<td>L2</td>
<td>10:30</td>
<td>130</td>
<td>0.3</td>
</tr>
<tr>
<td>t₆</td>
<td>H-121</td>
<td>L3</td>
<td>10:30</td>
<td>110</td>
<td>0.2</td>
</tr>
<tr>
<td>t₇</td>
<td>Y-324</td>
<td>L4</td>
<td>10:30</td>
<td>90</td>
<td>0.5</td>
</tr>
<tr>
<td>t₈</td>
<td>X-827</td>
<td>L4</td>
<td>10:20</td>
<td>105</td>
<td>0.35</td>
</tr>
<tr>
<td>t₉</td>
<td>X-827</td>
<td>L5</td>
<td>10:15</td>
<td>90</td>
<td>0.4</td>
</tr>
<tr>
<td>t₁₀</td>
<td>C-442</td>
<td>L5</td>
<td>10:10</td>
<td>120</td>
<td>0.3</td>
</tr>
<tr>
<td>t₁₁</td>
<td>C-442</td>
<td>L2</td>
<td>10:05</td>
<td>140</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Figure 1: A probabilistic relation

for top-k queries [22, 27, 12], but it has never been investigated for skylines.

In this paper we address the problems of defining and efficiently computing the skyline of a probabilistic relation. As a first step we provide a formal definition of skyline, which is based on a generalization to the probabilistic case of the concept of domination among tuples. The P-domination relationship we introduce to this purpose is formally grounded in order theory, in particular on the connection existing between strict partial orders (which lead to skylines) and linear orders (that yield top-1 results). We prove that our extension naturally satisfies all the basic skyline properties. Because P-domination is tightly related to the semantics used to linearly rank probabilistic tuples, the skyline we obtain is parametric in such semantics. This is to say that, whatever semantics for top-k queries one wants to adopt, our skyline definition will be always consistent with it, which is a remarkable property. For instance, if one adopts the “expected rank” semantics [12] (see Section 4.1), the skyline of the relation in Figure 1 will consist of tuples \( t_5 \) and \( t_7 \). Thus, only \( t_5 \) and \( t_7 \) could be top-1 results, i.e., attaining the lowest expected rank, should one use any scoring function that is monotone in the Time and Speed attributes. This is exemplified in Figure 2, where two such scoring functions are considered.

We then consider how to efficiently check P-domination and, consequently, to compute the skyline. From the order-theoretic definition of P-domination one should collect all the top-1 results by varying (in infinite ways) the scoring function used to linearly rank tuples. Even considering linear orders (rather than scoring functions) the problem, although finite, would have exponential complexity. We solve the problem by demonstrating how P-domination can
be checked without the need of explicitly considering neither scoring functions nor linear orders. The intuition behind our approach is that one can demonstrate that tuple $u$ $P$-dominates tuple $v$ if $u$ is ranked (in a probabilistic sense) better than $v$ even for a scoring function (or linear order) that maximally favors $v$ with respect to $u$. We demonstrate how this approach results in specific $P$-domination rules for different semantics.

For reducing the actual response time of skyline computation, we introduce a family of algorithms based on a common two-phase structure. The algorithms adopt a set of optimization strategies, such as pre-sorting tuples, spatial indexing, and $P$-domination rule ordering. Since such strategies are orthogonal to each other, they can be combined in a modular way. Finally, we extensively evaluate our algorithms on a variety of datasets, which shows the practical applicability of our approach.

Summarizing, our contributions are:

- We introduce a novel definition of domination among probabilistic tuples ($P$-domination);
- We prove that the skyline resulting from this definition satisfies all the properties that hold in the deterministic case;
- We show how $P$-domination can be efficiently checked, and detail the analysis for different alternative semantics;
- We provide a family of algorithms able to efficiently compute the sky-

<table>
<thead>
<tr>
<th>TID</th>
<th>$t$</th>
<th>$s$</th>
<th>$w_t = 0.5, w_s = 0.5$</th>
<th>$w_t = 0.7, w_s = 0.3$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>score</td>
<td>exp. rank</td>
</tr>
<tr>
<td>$t_1$</td>
<td>71.5</td>
<td>2.11</td>
<td>64.1</td>
<td>2.03</td>
</tr>
<tr>
<td>$t_2$</td>
<td>75</td>
<td>2.2275</td>
<td>65</td>
<td>2.1825</td>
</tr>
<tr>
<td>$t_3$</td>
<td>67.5</td>
<td>2.46</td>
<td>56.5</td>
<td>2.42</td>
</tr>
<tr>
<td>$t_4$</td>
<td>71</td>
<td>2.445</td>
<td>55.4</td>
<td>2.445</td>
</tr>
<tr>
<td>$t_5$</td>
<td>80</td>
<td>1.71</td>
<td>60</td>
<td>1.815</td>
</tr>
<tr>
<td>$t_6$</td>
<td>70</td>
<td>2.17</td>
<td>54</td>
<td>2.17</td>
</tr>
<tr>
<td>$t_7$</td>
<td>60</td>
<td>2</td>
<td>48</td>
<td>1.625</td>
</tr>
<tr>
<td>$t_8$</td>
<td>62.5</td>
<td>2.175</td>
<td>45.5</td>
<td>2.21</td>
</tr>
<tr>
<td>$t_9$</td>
<td>52.5</td>
<td>2.44</td>
<td>37.5</td>
<td>2.44</td>
</tr>
<tr>
<td>$t_{10}$</td>
<td>65</td>
<td>2.055</td>
<td>43</td>
<td>2.31</td>
</tr>
<tr>
<td>$t_{11}$</td>
<td>72.5</td>
<td>2.415</td>
<td>45.5</td>
<td>2.525</td>
</tr>
</tbody>
</table>

Figure 2: The effect of two scoring functions ($s(t) = w_t \cdot \text{minutes}(t, \text{Time}) + w_s \cdot t, \text{Speed}$) on tuples in Figure 1 and corresponding expected ranks.
line of a probabilistic relation under the relevant case of the expected rank semantics, demonstrating their applicability to a range of large datasets.

The rest of the paper is organized as follows: Section 2 introduces background definitions and surveys relevant related work. Section 3 contains the statement of the problem, showing how P-domination can be defined and checked for any pair of probabilistic tuples. In Section 4 we fully develop analyses covering all the most widely known cases, namely the expected rank, the expected score and the U-Topk/U-kRanks/Global-Topk semantics; in Section 5 we give algorithms for efficiently computing the skyline under expected rank and expected score; such algorithms are experimentally evaluated in Section 6; Finally, Section 7 concludes the paper.

2 Preliminaries

We model a probabilistic relation $R^p$ as a triple, $R^p = (R, p, G)$, where $R$ is a relation in the standard sense, i.e., a set of tuples, also called a deterministic relation; $p$ is a function that assigns to each tuple $u \in R$ a probability, $p(u) \in (0, 1]$; and $G$ is a set of generation rules that form a partition of $R$. Each $G \in G$, also called a group, defines a set of mutually exclusive tuples in $R$, whose overall probability does not exceed 1.\textsuperscript{2} We denote by $G_u$ the group of tuple $u$. A possible world $W$ of $R^p$ is a subset of tuples of $R$ such that $W$ does not contain two distinct tuples from a same group. The probability of possible world $W$ is computed as:

$$\Pr(W) = \prod_{u \in W} p(u) \times \prod_{G \cap W = \emptyset} (1 - \sum_{u \in G} p(u)) \quad (1)$$

The set of possible worlds of $R^p$ is denoted $W$.

Given a (deterministic) relation $R$ whose schema includes a set of numerical attributes $A = \{A_1, A_2, \ldots, A_d\}$, the skyline of $R$ with respect to $A$, denoted $\text{SKY}_A(R)$ or simply $\text{SKY}(R)$, is the set of undominated tuples in $R$. Assuming that on each attribute higher values are preferable, tuple $u$ (Pareto-)dominates tuple $v$, written $u \succ v$, iff it is $u.A_i \geq v.A_i$ for each $A_i \in A$ and there exists at least one attribute $A_j$ such that $u.A_j > v.A_j$. Thus:

$$\text{SKY}(R) = \{u \in R \mid \not\exists v \in R : v \succ u\} \quad (2)$$

When neither $u \succ v$ nor $v \succ u$ hold, we say that $u$ and $v$ are indifferent, written $u \sim v$.

\textsuperscript{2}This is the widely adopted $x$-relation model, introduced in the TRIO system [1].
A scoring function $s()$ on the attributes $A$ is **monotone** iff $u.A_i \geq v.A_i$ ($i = 1, \ldots, d$) implies $s(u) \geq s(v)$, and is also **domination-preserving** if $u \succ v$ implies $s(u) > s(v)$.$^3$ Note that, say, the max function is monotone, but not domination-preserving, e.g., $(3, 4) \succ (1, 4)$ yet $\max\{3, 4\} = \max\{1, 4\}$.

The skyline has three important properties: 1) By definition, it requires no parameter to be specified; 2) it is insensitive to attribute scales; and 3) it equals the union of top-1 results under all possible domination-preserving monotone scoring functions.$^4$ In the following we always implicitly assume that a monotone function is also domination-preserving.

### 2.1 Related Work

Over the last few years, different ranking semantics for answering top-$k$ queries in probabilistic databases have been introduced. Soliman et al. [22] propose two distinct semantics: $U$-$Topk$, which returns the most likely top-$k$ result set, considering the probabilities of the possible worlds of $R^p$; and $U$-$kRanks$, that, for each $i = 1, \ldots, k$, returns the tuple with the highest probability of being at rank $i$. Zhang and Chomicki [27] introduce the $Global$-$Topk$ semantics, which computes for each tuple $u$ the probability that $u$ is among the top-$k$ tuples in the possible worlds of $R^p$, and then returns the $k$ tuples with the highest probabilities.

Based on the observation that none of these semantics defines a ranking of tuples in the strict sense, i.e., the top-$k$ tuples need not to be a subset of the result of a top-$(k + 1)$ query, in [12] Cormode et al. proposed the **expected rank** semantics. Here one computes the rank (position) that tuple $u$ has in each of the possible worlds of $R^p$, and then takes the expected value of such ranks. The **expected score** semantics is also introduced in [12], in which case ranking is simply based on the product of score and probability. A discussion on the properties of the various ranking semantics can be found in [27, 12, 17].

The first work to consider skyline queries on probabilistic data has been [20]. There, the basic idea is to compute for each tuple $u$ the probability, $Pr_{Sky}(u)$, that $u$ is undominated, and then rank tuples based on these skyline **probabilities**.

The skyline probability of $u$, $Pr_{Sky}(u)$, is the overall probability of the

---

$^3$Domination-preserving monotone functions coincide with those that others call *strictly monotone in each argument* [15].

$^4$Although it is folklore that the skyline of $R$ equals the union of top-1 results under all possible monotone scoring functions, this is imprecise because of the non-deterministic nature of top-1 queries. For instance, given $R = \{(3, 4), (1, 4)\}$ and the max function, $(1, 4)$ might be (non-deterministically) returned as the top-1 result.
possible worlds $W$ in which $u$ is in the skyline of $W$. $\Pr_{\text{Sky}}(u)$ is derived by observing that $u \in \text{Sky}(W)$ iff $u \in W$, which occurs with probability $p(u)$, and no tuple $v \in W$ dominates $u$, i.e.:

$$\Pr_{\text{Sky}}(u) = p(u) \times \prod_{G \neq G_u} (1 - \sum_{t \in G \mid t > u} p(t))$$

Finally, the $p$-skyline of $R^p$, based on a probability threshold $p$, is defined as:

$$\text{Sky}(R^p; p) = \{u \mid \Pr_{\text{Sky}}(u) \geq p\}$$

It has to be observed that this approach is unable to preserve the basic skyline properties, since it requires an additional parameter (the $p$ threshold), and has no apparent relationship with the results of top-1 queries. Subsequent works on the subject have provided efficient algorithms to compute all skyline probabilities [3], and shown how to compute $p$-skylines on uncertain data streams [26].

More recently, Lin et al. have proposed the stochastic skyline operator [18]. Given two uncertain objects $U$ and $V$, each characterized by a discrete probability distribution, $U$ stochastically dominates $V$ iff, for each point $x$ in the attribute space, the cumulative probability of $U$ in $x$, $U.cdf(x)$, is at least equal to $V.cdf(x)$, and there exists a point $y$ such that $U.cdf(y) > V.cdf(y)$. This method, although not requiring any additional parameter, has some major limitations. First, testing stochastic domination is an NP-complete problem, thus the method does not scale well with the number of skyline attributes. Second, the stochastic skyline equals only a subset of possible top-1 results, namely those arising from the expectation of multiplicative scoring functions, i.e., $f(U) = E[\prod_{i=1}^{d} f_i(A_i)]$.

3 P-Domination and the Skyline of a Probabilistic Relation

In this section we provide a definition of skyline for probabilistic relations and prove that it preserves all the basic properties the skyline has in the deterministic case. We start by rewriting Equation 2 as:

$$\text{Sky}(R^p) = \{u \in R \mid \not\exists v \in R : v \succ_p u \}$$

where the only difference with the deterministic case is that $\succ$ is substituted by $\succ_p$. We call $\succ_p$ the probabilistic domination relationship, or P-domination.
for short. Note that $\succ_p$ is a binary relation in the standard sense, i.e., no probability at all is present in $\succ_p$.

In order to preserve all skyline properties, we define P-domination using principles of order theory. In order-theoretic terms, $\succ$ is a strict partial order, i.e., an irreflexive ($\forall u : u \not\succ u$) and transitive ($\forall u,v,t : u \succ v \land v \succ t \Rightarrow u \succ t$) binary relationship. A linear order $\succ$ is a strict partial order that is also connected, i.e., for any two distinct tuples $u$ and $v$, either $u \succ v$ or $v \succ u$.

A linear order $\succ$ is a linear extension of $\succ$ iff $u \succ v \Rightarrow u \succ_p v$, that is, $\succ$ is compatible with $\succ$. Notice that a linear extension $\succ$ of $\succ$ is what one obtains by ordering tuples with a monotone scoring function $s()$ and then breaking ties arbitrarily.

Let $\text{Ext}(\succ)$ denote the set of all linear extensions of $\succ$. A fundamental result in order theory, derived from Szpilrajn’s Theorem [23], asserts that any strict partial order $\succ$ equals the intersection of its linear extensions, i.e.:

$$\succ = \bigcap \{ \succ \mid \succ \in \text{Ext}(\succ) \}$$

This is the first ingredient needed to define P-domination.

Our second ingredient exploits the observation that a linear order $\succ$ on the tuples of $R$ can be used to define a corresponding linear order on the probabilistic tuples of $R^p$. Note that this is exactly what semantics for top-$k$ queries that define a ranking in the proper sense of the term, such as expected rank and expected score, do. Further, as it will be made clear at the end of this section, this also applies to the other semantics surveyed in Section 2.1 by considering for them the ranking induced on tuples when $k = 1$.

For the sake of generality, we can represent a semantics for top-$k$ (actually, top-1) queries with a probabilistic ranking function $\Psi$ that, given a linear order $\succ$ on the tuples of $R$ and a probability function $p$, yields a probabilistic linear order $\succ_p = \Psi(\succ, p)$ on the tuples of $R^p$. Under this view, the actual ranking of the probabilistic tuples is obtained by computing for each tuple $u$ a value $\psi_p(u)$, so that $u \succ_p v$ if $\psi_p(u) > \psi_p(v)$. In case of ties, i.e., $\psi_p(u) = \psi_p(v)$, we assume a domination-preserving tie-breaking rule, i.e., if $u \succ v$ then $u \succ_p v$, otherwise the tie can be arbitrarily broken.

We are now ready to define P-domination:

**Definition 1** (P-domination). Let $R^p = (R, p, \mathcal{G})$ be a probabilistic relation, and let $\succ$ be the Pareto-domination relationship on the tuples in $R$ when considering the skyline attributes $\mathcal{A}$. Let $\Psi$ be a probabilistic ranking function on $R^p$. For any two tuples $u$ and $v$ in $R^p$, we say that $u$ P-dominates $v$, written

To denote linear orders over tuples we use the symbol $\succ$ in place of the usual $>$, and reserve the latter for the standard order on real numbers.
The following theorem directly follows from Definition 1:

**Theorem 1.** For any probabilistic ranking function \( \Psi \), the corresponding P-dominance relationship \( \succ_p \) is a strict partial order.

**Proof.** Clearly, \( \succ_p \) is irreflexive. To prove that also transitivity holds, assume that \( u \succ_p v \) and \( v \succ_p t \) both hold. Given \( \Psi \), this implies that, for each linear extension \( \succ \) of \( \succ \) and associated probabilistic linear order \( \succ_p = \Psi(\succ, p) \), it is \( u \succ_p v \) and \( v \succ_p t \), thus \( u \succ_p t \) as required. \( \square \)

Next theorem extends to the probabilistic realm the fundamental property of the skyline to contain all and only the top-1 results of (domination-preserving) monotone scoring functions.

**Theorem 2.** Let \( \text{SKY}(R^p) \) be the skyline of \( R^p \), for a given probabilistic ranking function \( \Psi \). A tuple \( u \) belongs to \( \text{SKY}(R^p) \) iff there exists a monotone scoring function \( s() \) such that \( u \) is the top-1 tuple according to the probabilistic linear order \( \succ_p = \Psi(\succ, p) \), where \( \succ \) is the linear order induced by \( s() \) on \( R \).

**Proof.** (if) Assume that for some scoring function \( s() \), with induced linear order \( \succ \), tuple \( u \in R^p \) is the top-1 for the probabilistic ranking \( \succ_p = \Psi(\succ, p) \).

To prove that \( u \in \text{SKY}(R^p) \) we have to show that no tuple \( v \) P-dominates \( u \).

By contradiction, assume this is not the case, i.e., there exists \( v \) such that \( v \succ_p u \). In turn, this implies \( v \succ_p u \) for all probabilistic rankings obtained from a linear extension of \( \succ \), thus \( u \) can never be a top-1 tuple, a contradiction.
We prove that if \( u \in \text{Sky}(R_p) \), then there exists \( s() \), with induced linear order \( \succ \), such that \( u \) is the top-1 for the probabilistic ranking \( \succ_p = \Psi(\succ, p) \). By hypothesis, for any tuple \( v \) either \( u \) P-dominates \( v \) or the two are probabilistically indifferent, written \( u \sim_p v \). To prove the result we make use of Szpilrajn’s Theorem [23], that in our context asserts that if two tuples are indifferent, \( u \sim_p v \), then there exist two linear extensions \( \succ_{p,1} \) and \( \succ_{p,2} \) of \( \succ_p \) such that \( u \succ_{p,1} v \) and \( v \succ_{p,2} u \), respectively. By applying the theorem to all the tuples \( v \) that are probabilistically indifferent to \( u \) it is therefore possible to derive a probabilistic linear order in which \( u \) is the top-1 tuple, from which the result follows.

Further, as in the deterministic case, the skyline of \( R_p \) is insensitive to actual attribute values, rather it only depends on the relative ordering on each skyline attribute.

**Theorem 3.** Let \( R_p = (R, p, G) \) be a probabilistic relation, and let \( A \) be the skyline attributes. Let \( S_p = (S, p, G) \) be a probabilistic relation obtained from \( R_p \) as follows: \( S \) is obtained from \( R \) through an isomorphism \( \phi \) that preserves Pareto domination, i.e., \( \forall u, v \in R \) it is \( \phi(u) \succeq \phi(v) \) iff \( u \succeq v \); \( \forall u \in R \) it is \( p(\phi(u)) = p(u) \) (probabilities are preserved); \( \forall u, v \in R \) it is \( G_{\phi(u)} = G_{\phi(v)} \) iff \( G_u = G_v \) (groups do not change as well). Then, for any probabilistic ranking function \( \Psi \), it is \( \text{Sky}(R_p) = \text{Sky}(S_p) \).

**Proof.** Because of the theorem’s hypotheses and of the observation that actual attribute values are not used at all to define P-domination (and consequently the skyline of \( R_p \)), the result immediately follows.

Due to Theorems 2 and 3, and to the fact that no parameter at all is used, it is proved that our skyline definition benefits all the three properties the skyline has in the deterministic case.

We now explain how our skyline definition applies to those semantics, such as U-Topk, U-kRanks, and Global-Topk, for which the ranking of tuples varies with the value of \( k \). We have the following preliminary observation (see also [24]):

**Observation 1.** For any probabilistic relation \( R_p \), whose tuples are linearly ordered according to \( \succ \), the result of a top-1 query (i.e., \( k = 1 \)) computed according to the U-Top1, U-1Ranks, and Global-Top1 semantics is the same, and is given by the tuple \( u \) that maximizes the top-1 probability defined as:

\[
\Pr_{\succ}^{\text{top1}}(u) \overset{\text{def}}{=} p(u) \times \prod_{G \neq G_u} \left( 1 - \sum_{t \in G \atop G \succ u} p(t) \right)
\]
From above observation it is evident that, if for such semantics one takes \( \psi_m(u) = \Pr_{P_{\top_1}}(u) \), then all the results we have derived smoothly applies. In particular, Theorem 2 confirms that even for such semantics the skyline will equal the set of tuples that can be top-1 for some scoring function.

### 3.1 How to Check P-Domination

Definition 1 cannot be directly used to check P-domination, since it requires to enumerate all linear extensions of \( \succ \), and these can be exponential in the number of tuples.\(^6\) The approach we pursue, and that can be applied regardless of the specific ranking semantics \( \Psi \), does not require to materialize any linear extensions of \( \succ \) and therefore results in efficient (polynomial) algorithms for skyline computation.

Consider a linear extension \( \succ \to \succ \), and let \( \psi_m(u) \) be the numerical value that \( \Psi \) assigns to tuple \( u \). According to Definition 1, for \( u \succ v \) to hold it has to be \( \psi_m(u) > \psi_m(v) \) for all linear extensions \( \succ \to \succ \), that is:

\[
\min_{\succ \in \text{Ext}(\succ)} \left\{ \frac{\psi_m(u)}{\psi_m(v)} \right\} > 1 \quad (8)
\]

The key idea for efficiently checking the above inequality without enumerating all linear extensions of \( \succ \) is to determine which is the linear order that is the most unfavorable one for \( u \) with respect to \( v \). If \( \psi_m(u) > \psi_m(v) \) holds for this “extremal” order, then it will necessarily hold for all other orders compatible with \( \succ \).

Regardless of the specific ranking semantics \( \Psi \), there are two cases to consider:

**u \succ v:** When \( u \) dominates \( v \), we can restrict the analysis to those linear orders for which it is \( u \succ v \); among them, we determine how other tuples \( t \) that are indifferent to either \( u \) and \( v \) (or both) should be arranged in the linear order so as to minimize the ratio \( \psi_m(u)/\psi_m(v) \).

**u \not\succ v:** If \( u \) does not dominate \( v \), then the worst case for \( u \) and the best one for \( v \) corresponds to a linear order in which: 1) \( u \succ t \) only for those tuples \( t \) that \( u \) dominates, and 2) \( t' \succ v \) only for those tuples \( t' \) that dominate \( v \).

Summarizing, this approach will result in a set of P-domination rules that can be checked without the need of materializing any linear extension of \( \succ \).

\(^6\)If \( R_p \) consists of \( N \) pairwise indifferent tuples, then \( \text{Ext}(\succ) \) has size \( N! \), since each permutation is compatible with \( \succ \).
4 P-domination Rules

In this section we derive specific P-domination rules for the following semantics: expected rank, expected score, U-Topk, U-kRanks, and Global-Topk.

4.1 P-domination with Expected Ranks

According to [12], the result of a top-k query on a probabilistic relation $R^p$ is based on the concept of expected rank. Given a linear order $\succ$ on the tuples of $R$, the rank of $u$ in a possible world $W$ with $|W|$ tuples is the number of tuples in $W$ that precedes $u$, that is:

$$\text{rank}_{W,\succ}(u) = \begin{cases} \{v \in W \mid v \succ u\} & \text{if } u \in W \\ |W| & \text{otherwise} \end{cases}$$

Therefore, the expected rank of a tuple $u$ is:

$$ER_{\succ}(u) = \sum_{W \in W} \text{rank}_{W,\succ}(u) \times \Pr(W)$$

As in [12], we consider that in case two tuples have a same expected rank value, a (domination-preserving) tie-breaking rule is applied so that expected ranks define a linear order. Let $\succ_p$ be such linear order, i.e.,: $u \succ_p v$ iff $ER_{\succ}(u) < ER_{\succ}(v)$. Thus, for this semantics it is $\psi_{\succ}(u) = -ER_{\succ}(u)$.

The expected rank of a tuple $u$ can be expressed as [12]:

$$ER_{\succ}(u) = p(u) \times \sum_{t \succ_u, t \notin G_u} p(t) + (1 - p(u)) \times \left( \sum_{t \in G_u, t \neq u} p(t) + \frac{\sum_{t \in G_u, t \neq u} p(t)}{1 - p(u)} \right)$$

where the first term is the expected rank of $u$ in a possible world in which $u$ appears, whereas the second term is the expected size of a world that does not contain $u$.

Let $H_{G,\succ}(u) = \sum_{t \succ_u, t \notin G_u} p(t)$ be the overall probability of those tuples not in $G_u$ that are better than $u$ according to $\succ$. Further, let $P$ be the overall probability of all the tuples in $R^p$, $P = \sum_{t \in R^p} p(t)$, and $P_{G_u} = \sum_{t \in G_u, t \neq u} p(t)$. Then, $ER_{\succ}(u)$ can also be compactly written as:

$$ER_{\succ}(u) = p(u) \times H_{G,\succ}(u) + (1 - p(u)) \times (P - P_{G_u} - p(u)) + P_{G_u}$$

According to Definition 1, tuple $u$ P-dominates $v$ iff it is $\max_{\succ \in \text{EXT}(\succ)} \{ER_{\succ}(u)/ER_{\succ}(v)\} < 1$, i.e., $ER_{\succ}(u) < ER_{\succ}(v)$ for any $\succ \in$
\[Ext(\succ).\] This can be equivalently written as:

\[
p(u) \times H_{G,\succ}(u) + (1 - p(u)) \times (P - P_{G_u} - p(u)) + P_{G_u} < \\
p(v) \times H_{G,\succ}(v) + (1 - p(v)) \times (P - P_{G_v} - p(v)) + P_{G_v}
\]

After defining \(P_{u,v} = P - p(u) - p(v)\) we first obtain:

\[
p(u) \times H_{G,\succ}(u) + (1 - p(u)) \times (P_{u,v} + p(u) - P_{G_u}) + P_{G_u} < \\
p(v) \times H_{G,\succ}(v) + (1 - p(v)) \times (P_{u,v} + p(u) - P_{G_v}) + P_{G_v}
\]

Rearranging terms and simplifying, this can be equivalently written as:

\[
p(u) \times (H_{G,\succ}(u) - P_{u,v} + P_{G_u} - 1) < p(v) \times (H_{G,\succ}(v) - P_{u,v} + P_{G_v} - 1)
\]

Since both sides are negative, changing the sign it is obtained:

\[
p(u) \times (P_{u,v} + 1 - H_{G,\succ}(u) - P_{G_u}) > p(v) \times (P_{u,v} + 1 - H_{G,\succ}(v) - P_{G_v})
\]

which leads to the definition of P-domination for the expected rank semantics as:

\[
u \succ_p v \Leftrightarrow \max_{\succ \in Ext(\succ)} \left\{ \frac{P_{u,v} + 1 - H_{G,\succ}(v) - P_{G_v}}{P_{u,v} + 1 - H_{G,\succ}(u) - P_{G_u}} \right\} \quad (9)
\]

The two cases to consider for checking the validity of Inequality 9 are dealt with as follows. For both here we describe the general case in which \(u\) and \(v\) belong to different groups first, postponing the (easier) case \(G_u = G_v\).

**u \succ v**: The first rule we derive is sufficient (but not necessary) for P-domination to occur, and has the advantage that it can be more efficiently checked than the other we derive.

Since \(u \succ v\), it is \(H_{G,\succ}(v) \geq H_{G,\succ}(u) + p(u) - P_{G_u}\), which is obtained by assuming a worst-case scenario for \(u\) in which all tuples \(t \in G_v\) other than \(v\) dominate \(u\). Substituting in Inequality 9 it is obtained:

\[
u \succ v \wedge \frac{p(u)}{p(v)} \geq 1 \wedge \frac{p(u)}{P_{G_u}} \geq 1 \quad (ER:Rule 1)
\]

Note that in case neither inequality holds strictly it is still safe to say that \(u \succ_p v\), since we assume a domination-preserving tie-breaking rule.
In case Rule 1 is not satisfied, a preliminary key observation is that, for any linear order \( m \) that extends \( \succ \), it is \( H_G,\succ(u) \in [H_G^-(u), H_G^+(u)] \), where the two bounds are respectively defined as:

\[
H_G^-(u) = \sum_{t \succ u, t \notin G_u} p(t) \quad H_G^+(u) = \sum_{u \prec t, t \notin G_u} p(t)
\]

Clearly, \( H_G^-(u) \) is the best possible case for \( u \), since only those tuples from other groups that dominate \( u \) are also better than \( u \) according to \( m \), whereas the worst case for \( u \) arises when \( u \) is better only of those tuples of other groups that it dominates.

Now, the ratio in the right-hand side of Inequality 9 can be maximized by adequately arranging those tuples \( t \) that are indifferent to either \( u \) or \( v \) (or both). Since \( u \succ v \), and \( \succ \) is transitive, only three cases can occur:

**t \sim u, t \succ v:** In order to favor \( v \) with respect to \( u \) it has necessarily to be \( t \succ u \) whenever \( t \notin G_u \). Let \( IB(u, v) = \sum_{t \sim u, t \succ v, t \notin G_u} p(t) \) stand for the mass of probability of such tuples.

**t \sim v, u \succ t:** For the same reason it has to be \( t \prec v \).

**t \sim u, t \sim v:** To analyze this case, first observe that setting \( t \sim u \sim v \) when \( t \notin G_u \) would penalize only \( u \). Let \( II_{G_v}(u, v) = \sum_{t \sim u, t \sim v, t \in G_v} p(t) \) be the total mass of probability of such tuples. At this point the ratio in Inequality 9 would be written as:

\[
\frac{N(u, v)}{D(u, v)} = \frac{P_{u,v} + 1 - H_G^-(v) - P_{G_v}}{P_{u,v} + 1 - H_G^-(u) - 1B(u, v) - II_{G_v}(u, v) - P_{G_u}}
\]

Consider now those tuples \( t \notin G_u \cup G_v \), whose total mass of probability is \( \Delta(u, v) = \sum_{t \sim u, t \sim v, t \notin G_u \cup G_v} p(t) \). The two alternatives to consider are either \( t \succ u \succ v \) or \( u \succ v \succ t \). The choice actually depends on the value of \( N(u, v)/D(u, v) \):

If \( N(u, v) < D(u, v) \), then setting \( t \succ u \succ v \) would lead to the ratio \( (N(u, v) - \Delta(u, v))/(D(u, v) - \Delta(u, v)) \), which is lower than \( N(u, v)/D(u, v) \).

On the other hand, when \( N(u, v) > D(u, v) \), then \( (N(u, v) - \Delta(u, v))/(D(u, v) - \Delta(u, v)) > N(u, v)/D(u, v) \), thus the case \( t \succ u \succ v \) is the one to consider.

Finally, when \( N(u, v) = D(u, v) \) Inequality 9 degenerates to \( p(u) > p(v) \).
Putting all together, the 2nd P-domination rule is:

\[
\begin{align*}
u \succ v \land \frac{p(u)}{p(v)} & \geq \begin{cases} 
\frac{N(u, v)}{D(u, v)} & \text{if } \frac{N(u, v)}{D(u, v)} \leq 1 \\
\frac{N(u, v) - \Delta(u, v)}{D(u, v) - \Delta(u, v)} & \text{otherwise}
\end{cases}
\end{align*}
\]

(ER:Rule 2)

where equality is again due to the domination-preserving rule for breaking ties.

\(u \not\succ v\): When \(u\) does not dominate \(v\), P-domination can still occur. In this case it is immediate to see that the ratio in Inequality 9 is maximized by setting \(H^-_G(v) = H^-_G(u)\), thus:

\[
\begin{align*}
u \not\succ v \land \frac{p(u)}{p(v)} & > \frac{P_{u,v} + 1 - H^-_G(v) - P_{G_v}}{P_{u,v} + 1 - H^-_G(u) - P_{G_u}}
\end{align*}
\]

(ER:Rule 3)

We conclude the analysis for the expected rank semantics by detailing P-domination rules for mutually exclusive tuples, i.e., tuples belonging to a same group \(G\).

**Rule 1:** Since \(u \succ v\), now it is \(H_G(v) \geq H_G(u)\), thus the first P-domination rule reduces to:

\[
\begin{align*}
u \succ v \land \frac{p(u)}{p(v)} & \geq 1
\end{align*}
\]

(ER:Rule 1-sg)

Note that when \(p(u) = p(v)\), \(u\) and \(v\) would have the same expected rank, in which case \(u \succ_p v\) holds by the domination-preserving tie-breaking rule.

**Rule 2:** The only difference with the scenario analyzed in Section 4.1 is that, since the quantity \(H_G(v, u) = \sum_{t \sim u \sim v, t \in G_v} p(t)\) now does not contribute to \(H_G(u)\), this term vanishes from the denominator in Equation 10. Apart from this the analysis proceeds as in the case of different groups.

**Rule 3:** This rule applies unchanged.

**Example 1.** Table 1 lists the probabilities of the tuples in Figure 1, whose overall probability is \(P = 2.7\), together with their \(H^-_G\) and \(H^+_G\) bounds.

As an example of how bounds are computed consider tuple \(t_8\). Since \(t_8\) is dominated only by \(t_4\), \(t_5\), and \(t_6\), it is \(H^-_G(t_8) = p(t_4) + p(t_5) + p(t_6) = 0.6\). The only tuple dominated by \(t_8\) is \(t_9\), thus \(H^+_G(t_8) = P - p(t_8) - p(t_9) = 1.95\).
Table 1: Probabilities and bounds for the dataset in Figure 1

<table>
<thead>
<tr>
<th>tuple</th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
<th>$t_5$</th>
<th>$t_6$</th>
<th>$t_7$</th>
<th>$t_8$</th>
<th>$t_9$</th>
<th>$t_{10}$</th>
<th>$t_{11}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>probability</td>
<td>0.2</td>
<td>0.15</td>
<td>0.1</td>
<td>0.3</td>
<td>0.2</td>
<td>0.5</td>
<td>0.35</td>
<td>0.4</td>
<td>0.3</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>$H^-_G$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>1.05</td>
<td>0.6</td>
<td>1.55</td>
<td>0.3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$H^+_G$</td>
<td>1.6</td>
<td>1.55</td>
<td>1.55</td>
<td>0.85</td>
<td>0.55</td>
<td>1.25</td>
<td>1.8</td>
<td>1.95</td>
<td>1.95</td>
<td>2.3</td>
<td>2.3</td>
</tr>
<tr>
<td>$Pr_{Sky}$</td>
<td>0.2</td>
<td>0.15</td>
<td>0.1</td>
<td>0.1</td>
<td>0.3</td>
<td>0.12</td>
<td>0.144</td>
<td>0.168</td>
<td>0.0576</td>
<td>0.21</td>
<td>0.1</td>
</tr>
</tbody>
</table>

A case to which Rule 1 applies concerns tuples $t_5$ and $t_6$, since $t_5 \succ t_6$, $0.3 = p(t_5) \geq p(t_6) = 0.2$, and $0.3 = p(t_5) \geq P_{G_{t_5}} = 0.1$. Rule 1 allows tuples $t_3$, $t_6$, $t_9$, and $t_{10}$ to be immediately discarded, since they are P-dominated by $t_2$, $t_5$, $t_7$, and $t_5$ again, respectively. Rule 2 is used to show that $t_5 \succ_p t_8$ and $t_5 \succ_p t_9$. A case to which Rule 3 applies is to prove that $t_7 \succ_p t_2$ holds. Notice that not only $t_7 \not\succ t_2$, but $t_2 \succ t_7$ holds in the deterministic case. An exhaustive analysis shows that for this example it is $Sky(R^p) = \{t_5, t_7\}$.

The last row of Table 1 shows the skyline probabilities, $Pr_{Sky}$, of all tuples (see also Equation 3 in Section 2.1). Although $t_5$, which belongs to both $Sky(R)$ and $Sky(R^p)$, has also the highest skyline probability, this is not the case for tuple $t_7$, which would be ranked only 6-th according to the method of [20]. In general, there is no clear correlation among P-domination and $Pr_{Sky}(u)$, i.e., a tuple might have a high skyline probability and at the same time to be P-dominated by several other tuples.

### 4.2 P-domination with Expected Scores

In [12], the case where tuples are ranked by their expected score, $ES(u) = s(u) \times p(u)$, is also considered. For this probabilistic ranking function (the only one that explicitly depends on tuples’ scores) it is immediate to derive that:

\[
\begin{align*}
  u \succ v \land \frac{p(u)}{p(v)} \geq 1
\end{align*}
\]  

(ES:Rule 1)

This holds since, for any scoring function $s()$, it has to be $s(u) \times p(u) > s(v) \times p(v)$, i.e., $p(u)/p(v) > s(v)/s(u)$. If $u \not\succ v$, then for any given value of the ratio $p(u)/p(v)$ the right-hand side can be made arbitrarily large and P-domination does not hold. If $u \succ v$, it is $s(v)/s(u) = 1 - \epsilon$, with $\epsilon > 0$ arbitrarily small, from which the result follows.

It is remarkable that, although ranking by expected scores is highly dependent on actual score values, the skyline of $R^p$ is, even in this case, insensitive
to attribute values. A major pitfall of expected score ranking is that it completely ignores correlation among tuples, and this obviously propagates to skylines. Further, since the above is the only P-domination rule, the skyline obtained from the expected score semantics will be always larger than the skyline in the deterministic case (since $u \succ v$ is necessary for P-domination to hold).

### 4.3 P-domination with U-Topk, U-kRanks, and Global-Topk

As already shown in Section 3, for all the U-Topk, U-kRanks, and Global-Topk semantics the result of a top-1 query is the tuple that maximizes the top-1 probability $\Pr_{\top_1}^u(u)$ defined by Equation 7. It is therefore possible to define P-domination for the above semantics by setting $\psi(u) = \Pr_{\top_1}^u(u)$ into Equation 8.

We first define $Q_G(u) = \prod_{G \neq G_u} \left(1 - \sum_{t \succ u, t \in G} p(t)\right)$. Then, $\Pr_{\top_1}^u(u)$ can be compactly written as:

$$\Pr_{\top_1}^u(u) = p(u) \times Q_G(u)$$

Thus, according to Definition 1, we obtain the definition of P-domination between tuples $u$ and $v$ as:

$$u \succ_p v \iff \frac{p(u)}{p(v)} > \max_{\succ \in \Ext(\succ)} \left\{ \frac{Q_G(v)}{Q_G(u)} \right\}$$

For deriving P-domination rules for this scenario, we follow similar arguments to those exploited in the case of the expected rank semantics.

**u \succ v:** To derive the first, sufficient but not necessary, rule we can assume a worst case scenario for $u$ where all tuples of $G_v$ other than $v$ dominate $u$, obtaining $Q_{G,v}(v) \leq Q_{G,v}(u)\frac{1 - p(u)}{1 - \Pr_G}$. This, substituted into Equation 11, gives us the following rule:

$$u \succ v \land \frac{p(u)}{p(v)} \geq \frac{1 - p(u)}{1 - \Pr_G}$$

(Top1:Rule 1)

If Rule 1 does not hold, we can still bound $Q_{G,v}(u)$ by observing that the best possible order for $u$ is the one where $u$ is preceded by only those tuples that dominate $u$, while the worst order is the one that puts $u$ after all
tuples not dominated by $u$. Thus, it is $Q_{G,\succ}(u) \in [Q_G^-(u), Q_G^+(u)]$, where the bounds are defined as follows:

$$Q_G^-(u) = \prod_{G \not\supseteq G_u} \left(1 - \sum_{u \neq t, t \in G} p(t)\right) \quad Q_G^+(u) = \prod_{G \not\subseteq G_u} \left(1 - \sum_{t > u, t \in G} p(t)\right)$$

As in Section 4.1, in order to maximize the ratio in the right side of Equation 11 one should only arrange those tuples $t$ that are indifferent to $u$ or $v$ (or both). If $u$ is to be penalized, we should maximize the number of tuples preceding it. To this end, the tuples $t \not\in G_u, t \sim u, t > v$ and $t \in G_v, t \sim u, t \sim v$ can be all ordered so as to precede $u$, since they either precede also $v$ or do not contribute to its top-1 probability. Consider now those tuples $t \not\in G_v : t \sim u, t \sim v$. The two alternatives to consider are either $t > u > v$ or $u > v > t$. To this end we first note that, since $u > v$, it is $\sum_{t > u} p(t) \leq \sum_{t > v} p(t)$, thus having $t > u > v$ would actually lower the ratio $(1 - \sum_{t \not\in G_u \cup G_v} p(t))/(1 - \sum_{t \in G_u \cap G_v} p(t))$, leading to an overall lower value of the right side of Equation 11. Thus, by defining $B_{\mathcal{IB}}(u, v)$ as:

$$B_{\mathcal{IB}}(u, v) \overset{\text{def}}{=} \left(1 - \sum_{(t \sim u \lor (t > u \land t \sim v)) \lor (t \sim u \land t > v)} p(t)\right) \times \prod_{G \not\subseteq G_u \lor G \not\supseteq G_v} \left(1 - \sum_{(t \sim u \lor (t > u \land t \sim v)) \lor (t \sim u \land t > v)} p(t)\right)$$

the second P-domination rule is:

$$u \succ v \land \frac{p(u)}{p(v)} \geq \frac{Q_G^+(v)}{B_{\mathcal{IB}}(u, v)} \quad \text{(Top1:Rule 2)}$$

$u \not\supseteq v$: It is easy to see that, since now it could happen that $v > u$, it is:

$$u \not\supseteq v \land \frac{p(u)}{p(v)} > \frac{Q_G^+(v)}{Q_G^-(u)} \quad \text{(Top1:Rule 3)}$$

Finally, we derive P-domination rules for the case of tuples belonging to a same group.

**Rule 1:** Since $u \succ v$, now it is $Q_{G,\succ}(v) \leq Q_{G,\succ}(u)$, thus the first P-domination equals the rule expressed by Equation ER:Rule 1-sg, i.e.:

$$u \succ v \land \frac{p(u)}{p(v)} \geq 1 \quad \text{(Top1:Rule 1-sg)}$$
Rule 2: The only difference with the previous case is that now \( B(u,v) \) should not include tuples in \( G_u \), i.e., the following value should be used in Equation Top1:Rule 2:

\[
B(u,v) = \prod_{G \neq G_u} \left( 1 - \sum_{\left( t \succ v \lor (t \sim u \land t \succ v) \right) \cap G} p(t) \right)
\]

Rule 3: This rule applies unchanged.

## 5 Algorithms

In this section we introduce efficient algorithms for computing the skyline of a probabilistic relation \( R^p \) consisting of \( N \) tuples. Since our current implementation only covers the expected rank and the expected score semantics, in the following we will not consider the U-Topk, U-kRanks, and Global-Topk semantics, whose P-domination rules have been defined in Section 4.3.

We first cover the simpler case of the expected score semantics. In this scenario, if we consider the probability of a tuple as a new attribute, the P-domination rule reported in Section 4.2 is equivalent to the definition of (deterministic) domination on the \( A \cup \{ p \} \) attributes (see Section 2). It immediately follows that any deterministic skyline algorithm (like BNL [6], SFS [11], or SaLSa [4]) that considers the set of attributes \( A \cup \{ p \} \) can be used for computing \( \text{SKY}(R^p) \). Therefore, it is immediately derived that the skyline of \( R^p \) can be computed in \( O(N^2) \) time.

Turning to considering the expected rank semantics, it is important to first characterize the time complexity of the three P-domination rules described in Section 4.1:

- In the worst case, evaluating Rule 1 over all pairs of tuples requires \( O(N^2) \) time. This clearly holds when the probability of each group is known in advance, but it remains true even if group probabilities need to be determined (since this can be done before actually starting to compare tuples, see also Section 5.1).

- Checking Rule 3 on a pair of tuples apparently has a complexity of \( O(N) \), because of the quantities \( H^-_G(v) \) and \( H^+_G(u) \); however, such values do not depend on the actual pair of tuples being compared, and thus can be pre-computed for each tuple, which collectively requires
$O(N^2)$ time. Thus, a single evaluation of Rule 3 is a constant time, $O(1)$, operation and evaluating Rule 3 on all pairs of tuples requires $O(N^2)$ time.

- A single evaluation of Rule 2 requires $O(N)$ time, because of the quantities $IB(u,v)$, $HG_v(u,v)$, and, possibly, $\Delta(u,v)$. Unless one can afford a $O(N^2)$ storage overhead for book-keeping (which we do not consider here), such quantities cannot be computed in advance because they depend on both $u$ and $v$.

Therefore, because of Rule 2, computing the skyline of $R^p$ requires $O(N^3)$ time in the worst case.

Based on above observations, our algorithms are designed around a common 2-phase template, shown in Algorithm 1:

**Algorithm 1 2-phase Algorithm template**

**Input:** probabilistic relation $R^p$

**Output:** $\text{Sky}(R^p)$, the skyline of $R^p$

**Phase I** (bounds computation and low-cost pruning)

1.a) determine whether tuples have to be sorted or not; compute basic groups’ information

1.b) pre-compute $H^-_G(u)$ and $H^+_G(u)$ for each tuple $u$, possibly applying some $P$-domination rule

**Phase II** (skyline computation)

2.a) determine whether a spatial index has to be built or not

2.b) determine how to apply the remaining $P$-domination rules

The common rationale of our algorithms is to first pre-compute, for each tuple $u$, the bounds $H^-_G(u)$ and $H^+_G(u)$ (Phase I.b). While doing this, one can also consider to apply Rule 1 to immediately discard some tuples, thus avoiding to compute their bounds. Pre-sorting tuples (Phase I.a) can turn to be beneficial both for speeding up bounds computation, and for allowing a modified version of Rule 3 to be also applied (see Section 5.1).

In the second phase we actually compute the skyline of $R^p$. Due to the high cost of evaluating Rule 2, here we consider the possibility of a 2-scan approach (Phase II.b), the first of which only involves Rule 1 (if not already applied) and Rule 3. To efficiently allow for testing Rule 2, in Phase II.a we also consider the possibility of building on-the-fly a spatial index.

Table 2 summarizes the options available for the two phases, together with symbols that will be used in graphs in Section 6.
Table 2: Alternatives available for the 2-phase Algorithm

<table>
<thead>
<tr>
<th>Phase</th>
<th>symbol</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I.a</td>
<td>unsrt</td>
<td>no sort</td>
</tr>
<tr>
<td></td>
<td>srt</td>
<td>uses topological sort</td>
</tr>
<tr>
<td>I.b</td>
<td>∅</td>
<td>no P-domination rule used</td>
</tr>
<tr>
<td></td>
<td>R1</td>
<td>uses Rule 1 only</td>
</tr>
<tr>
<td></td>
<td>R1R3⁺</td>
<td>uses Rules 1 and 3⁺</td>
</tr>
<tr>
<td>II.a</td>
<td>noidx</td>
<td>no index</td>
</tr>
<tr>
<td></td>
<td>idx</td>
<td>uses a spatial index (R-tree/aR-tree)</td>
</tr>
<tr>
<td>II.b</td>
<td>1scan</td>
<td>single scan using all Rules</td>
</tr>
<tr>
<td></td>
<td>2scans</td>
<td>1st scan with Rules 1 and 3, 2nd scan with Rule 2</td>
</tr>
</tbody>
</table>

5.1 Phase I

The main purpose of Phase I is to compute the $H^-_G$ and $H^+_G$ bounds.

The basic case to consider is the one where no particular order is imposed on tuples in $R^p$. In this case, the first action performed by Phase I.a in Algorithm 1 is to build an hash table, called groupsHT. Each entry of groupsHT corresponds to a specific group $G$, containing (backward) links to all tuples in $G$ and the sum of their probabilities, i.e., $\sum_{t \in G} p(t)$. Assuming direct access to groupsHT is in $O(1)$, computing the probabilities of all groups requires $O(N)$ time. Besides being used whenever group probabilities are needed to check P-domination rules, groupsHT is the key for the efficient implementation of Rule 2, as detailed in Section 5.2. Figure 4 shows how groupsHT looks like for the data shown in Figure 1.

In the unordered scenario, a naïve nested-loops is the simplest way to compute the bounds. This basic schema can be improved if, when computing $H^-_G(v)$ and $H^+_G(v)$ for a tuple $v$, we halt as soon as we find a tuple $u$ that P-dominates $v$ by using Rule 1.

An alternative implementation of Phase I.a is to perform a topological sort of the tuples in $R^p$ so that, if $u \succ v$, then $u$ will precede $v$ in the resulting order. Sorting tuples is executed before building the groupsHT hash table (so as to avoid updating backward links). Pre-sorting tuples has several benefits:

- $H^-_G(u)$ can be computed by just looking at tuples $t$ that precede $u$. 
Similarly, the computation of $H^+_G(u)$ needs to look only at tuples $t$ that follow $u$. To see why this is the case, observe that $H^+_G(u)$ can be equivalently expressed as

$$H^+_G(u) = P - (P_{G_u} + p(u)) - \sum_{u \succ t \atop t \not\in G_u} p(t) \quad (12)$$

where $P$ is the overall probability of all tuples in $R^p$ and $P_{G_u} + p(u)$ is the probability of all tuples in $G_u$. Since $u \succ t$ implies that $t$ follows $u$, the assert holds.

It is also evident that Rule 1 needs to be checked to determine if $u \succ_p v$ only if $u$ precedes $v$.

Finally, a topological sort of tuples allows for an approximate version of Rule 3 (Rule 3+) to be applied. Let us define $\overline{H^+_G}(u;v)$ as the running value of $H^+_G(u)$ at the time $v$ is accessed (this value will equal $H^+_G(u)$ when all tuples have been scanned), i.e.:

$$\overline{H^+_G}(u;v) = P - (P_{G_u} + p(u)) - \sum_{u \succ t \atop t \text{ precedes } v} p(t)$$

Notice that the sum equals 0 if $v$ precedes $u$. Since $\overline{H^+_G}(u;v) \geq H^+_G(u)$, then it is possible to bound from above the ratio in the right-hand side of Rule 3, which leads to:

$$u \neq v \land \frac{p(u)}{p(v)} > \frac{P_{u,v} + 1 - \overline{H}^-_G(v) - P_{G_v}}{P_{u,v} + 1 - H^+_G(u;v) - P_{G_u}} \quad \text{(ER:Rule 3+)}$$
Note that, as with Rule 1, Rule 3+ is only a sufficient condition for P-domination.

Algorithm 2 shows how Phase I would look like when topological sort is applied and Rules 1 and 3+ are used to filter out tuples. Tuples that survive after this phase are called candidates, with \( C^p \) denoting the set of all candidates.

**Algorithm 2 Phase I with Rules 1 and 3+**

**Input:** probabilistic relation \( R^p \)

**Output:** bounds \( H^-_G(u) \) and \( H^+_G(u) \) for each candidate tuple \((u \in C^p)\)

1: topologically sort \( R^p \)  
2: for all tuples \( u \in R^p \) do  
3: \( H^-_G(u) \leftarrow 0, \ H^+_G(u) \leftarrow P - (P_{G_u} + p(u)) \)  
4: \( u \_filtered \leftarrow \text{false} \)  
5: for all tuples \( v \in R^p \) preceding \( u \) do  
6: if \( v >_p u \) [Rule 1] then \( u \_filtered \leftarrow \text{true} \)  
7: if \( v \_filtered = \text{false} \land G_u \neq G_v \) then  
8: \( H^+_G(v) \leftarrow H^+_G(v) - p(u) \)  
9: if \( u \_filtered = \text{false} \land G_u \neq G_v \) then  
10: \( H^-_G(u) \leftarrow H^-_G(u) + p(v) \)  
11: else if \( v >_p u \) [Rule 3+] then \( u \_filtered \leftarrow \text{true} \)  
12: if \( u >_p v \) [Rule 3+] then \( v \_filtered \leftarrow \text{true} \)  

---

**5.2 Phase II**

In Phase II all P-domination rules that have not been applied in Phase I are used to filter out those candidate tuples in \( C^p \) that are not part of \( \text{Sky}(R^p) \).

Two main issues are to be addressed in this phase:

1. whether the rules are applied together (i.e., at the same time) or not, and

2. how to efficiently evaluate Rule 2, which is the most costly one.

As to the first point, we consider the possibility of a 2-scan approach: in the first scan of candidate tuples, only Rules 1 and 3 are applied (Rule 1 is skipped if it was already checked during Phase I), whereas the second scan applies Rule 2 to unfiltered tuples. The rationale for this approach is that, given the bounds computed in Phase I, Rule 3 has \( O(1) \) complexity, whereas a single evaluation of Rule 2 still requires \( O(N) \) time, thus it is
worthwhile to delay checking Rule 2 as much as possible. When tuples have been topologically sorted, this strategy indeed guarantees that Rule 2 is applied to the minimum number of pairs of tuples.

**Lemma 1.** If $R^p$ is topologically sorted and Rule 2 is applied only after both Rules 1 and 3 have been tested on all pairs of tuples, then it will never compare two tuples none of which is part of the skyline of $R^p$.

**Proof.** By contradiction, assume that Rule 2 is tested on tuples $u$ and $v$, none of which belongs to $\text{Sky}(R^p)$. Without loss of generality, assume that $u \succ v$. By hypothesis, there exists a skyline tuple $t$ that $P$-dominates $u$ and this $P$-domination is due to Rule 2, thus it is $t \succ u$. Since $R^p$ is topologically sorted, $t$ precedes $u$, thus $u$ will be dropped before $v$ is read, proving the assert. \hfill $\blacksquare$

The evaluation of Rule 2 for tuples $u$ and $v$, with $u \succ v$, needs quantities that depend on both $u$ and $v$, thus they cannot be pre-computed in Phase I. We show here how Rule 2 can be rewritten so as to guarantee its efficient computation. To this end, the quantities $N(u, v)$, $D(u, v)$, and $\Delta(u, v)$ are rewritten so as to avoid searching tuples that are indifferent to a given tuple; the rationale for this is to favor algorithms using either a spatial access method indexing the tuples in $R^p$ or a topological sort of tuples in $R^p$. In the former case, retrieving all tuples dominated by a tuple $u$ or that dominate $u$ requires issuing a single window query, while accessing all tuples that are indifferent to $u$ amounts to the resolution of $2^{d-1}$ window queries, where $d$ is the number of skyline attributes; in the latter case, tuples that dominates $u$ are guaranteed to precede $u$ in the input relation.

**Lemma 2.** The quantities $D(u, v)$, $D(u, v) - \Delta(u, v)$, and $N(u, v) - \Delta(u, v)$ that appear in Rule 2 can be equivalently rewritten as, respectively (for simplicity, we detail only the case when $G_u \neq G_v$):

$$D(u, v) = P_{u,v} + 1 - P_{G_u} - H_{G}(v) + \sum_{u \succ t \succ v} p(t)$$

$$+ \left( \sum_{t \in G_u} p(t) - \sum_{t \in G_v} p(t) \right) - \sum_{u \succ t \succ v} p(t) - \sum_{t \sim u} p(t)$$

(13)
We first prove Equation 13: from Equation 10 we have the following derivation:

Proof. We first prove Equation 13: from Equation 10 we have the following derivation:

\[ D(u, v) - \Delta(u, v) = P_{u,v} + 1 + P_{G_u} - 2P_{G_u} - (p(u) - p(v)) \]

\[ - H_G^-(u) - \left( \sum_{u \rightarrow t \rightarrow v} p(t) - \sum_{t \in G_u} p(t) \right) + \left( \sum_{t \sim v} p(t) - \sum_{t \sim v} p(t) \right) \]

\[ + \left( \sum_{u \rightarrow t \rightarrow v} p(t) - \sum_{t \sim v} p(t) \right) + \left( \sum_{t \sim v} p(t) - \sum_{t \sim v} p(t) \right) \]

(14)

\[ N(u, v) - \Delta(u, v) = P_{u,v} + 1 - (p(u) - p(v)) - P_{G_u} - H_G^+(u) \]

\[ - \sum_{u \rightarrow t \rightarrow v} p(t) + \sum_{t \in G_u} p(t) - \sum_{u \rightarrow t \rightarrow v} p(t) + \sum_{t \sim v} p(t) + \sum_{t \sim v} p(t) \]

(15)

\[ D(u, v) = P_{u,v} + 1 - H_G(u) - IB(u, v) - II_{G_u}(u, v) - P_{G_u} \]

\[ = P_{u,v} + 1 - \sum_{t \sim u \in G_u} p(t) - \sum_{t \sim v \in G_v} p(t) - \sum_{t \sim u \in G_u} p(t) - P_{G_u} \]

\[ = P_{u,v} + 1 - \sum_{t \sim u \in G_u} p(t) - \left( \sum_{t \sim v \in G_v} p(t) - \sum_{t \sim v \in G_v} p(t) \right) - \sum_{t \sim u \in G_u} p(t) - P_{G_u} \]

\[ = P_{u,v} + 1 - \left( \sum_{t \sim v} p(t) - \sum_{t \sim v} p(t) + \sum_{t \sim v} p(t) - \sum_{t \sim v} p(t) \right) \]

\[ + \left( \sum_{u \rightarrow t \rightarrow v} p(t) - \sum_{u \rightarrow t \rightarrow v} p(t) \right) - \sum_{t \sim u \in G_u} p(t) - P_{G_u} \]

\[ = P_{u,v} + 1 - \left( \sum_{t \sim v} p(t) - \sum_{t \sim v} p(t) \right) + \sum_{t \sim v} p(t) - \sum_{t \sim v} p(t) \]

25
For Equation 14, we first rewrite $\Delta(u, v)$ as:

\[
\Delta(u, v) = \sum_{t \sim u} p(t) = \sum_{t \in G_v} p(t) + \sum_{u \sim t \in G_u} p(t) - \sum_{u \sim t \in G_v} p(t)
\]

\[
= P - p(v) - P_{G_u} - H^-(v) - \left( \sum_{u \sim t} p(t) - \sum_{u \sim t \in G_v} p(t) \right) + \sum_{t \sim u} p(t)
\]

\[
= \left( P - p(u) - P_{G_v} - \sum_{t \in G_u} p(t) \right) + p(u) - p(v) + P_{G_u} - P_{G_v} - H^-(v)
\]

\[
+ \sum_{u \sim t \in G_u} p(t) + \left( \sum_{u \sim t \in G_v} p(t) - \sum_{u \sim t \in G_u} p(t) \right) + \sum_{t \sim u} p(t)
\]

\[
= (H^+(u) - H^-(v)) + (p(u) - p(v)) + (P_{G_u} - P_{G_v})
\]

\[
+ \left( \sum_{u \sim t \in G_u} p(t) - \sum_{u \sim t \in G_v} p(t) \right) + \left( \sum_{u \sim t \in G_v} p(t) - \sum_{u \sim t \in G_u} p(t) \right) + \sum_{t \sim u} p(t)
\]

Where the last step exploits Equation 12. Then the assert can be proven as:

\[
D(u, v) - \Delta(u, v) = P_{G_v} + 1 - P_{G_u} - H^-(v) + \sum_{u \sim t \sim v} p(t)
\]
Finally, Equation 15 is derived as follows:

\begin{align*}
N(u, v) - \Delta(u, v) &= P_{u,v} + 1 - H_G^-(v) - P_{G_v} - (H_G^+(u) - H_G^-(v)) \\
&\quad - (p(u) - p(v)) - (P_{G_u} - P_{G_v}) - \left( \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) - \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) \right) \\
&\quad - \left( \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) - \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) \right) - \sum_{\substack{t \sim u \atop t \in G_v}} p(t) \\
&= P_{u,v} + 1 - (p(u) - p(v)) - P_{G_u} - H_G^+(u) - \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) \\
&\quad + \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) - \sum_{\substack{u \succ v \in G_v \atop t \in G_u}} p(t) + \sum_{\substack{t \sim u \atop t \in G_v}} p(t)
\end{align*}

Terms appearing in above Equations fall into one of the three following categories:
1. Terms that also appear in other rules and, as such, are already available, namely: \( P_{u,v}, P_G u, P_G v \), and the bounds \( H_G(v) \) and \( H_G^+(u) \).

2. Sums that extend over tuples of a single group (either \( G_u \) or \( G_v \)); these can be efficiently evaluated thanks to the groupsHT hash table, that allows direct retrieval of all tuples in a group.

3. The sum of probabilities of all tuples \( t \) that are dominated by \( u \) and dominate \( v \), i.e., \( \text{WndPr}(u,v) \triangleq \sum_{u \succ t \succ v} p(t) \).

For computing the “window sum of probabilities” \( \text{WndPr}(u,v) \) we consider three methods:

1. The first alternative is to use a spatial access method, able to efficiently solve window queries. In particular, we consider both R-trees [16] and aR-trees [19]. With an R-tree, computing \( \text{WndPr}(u,v) \) amounts to retrieve all tuples in a window (whose extreme vertices coincide with attribute values of \( v \) and \( u \)) and summing their probabilities. On the other hand, since an aR-tree is able to solve aggregate queries, it can directly provide the required value, with the additional advantage of avoiding accessing index nodes whose region is completely included within the search window. Obviously, both techniques pay the cost of building the index, unless this is already available.

2. If no index is used, yet \( Rp \) has been sorted in Phase I.a, all tuples contributing to \( \text{WndPr}(u,v) \) will follow \( u \) and precede \( v \) in the order.

3. Finally, if \( Rp \) is unordered, we necessarily have to iterate through all tuples in \( Rp \).

Algorithm 3 shows an instantiation of Phase II when 2 scans of candidates are performed, such tuples are unordered, Rule 1 has been already applied in Phase I, and an aR-tree is built over \( Rp \).

6 Experimental Evaluation

In this section we experimentally analyze the efficiency of our algorithms for computing the skyline of a probabilistic relation. Our experiments have been performed on both real and synthetic datasets. The real dataset we used consists in statistics about NBA players, while synthetic datasets were considered to study the effects of data dimensionality and cardinality, as well as those due to different data and probability distributions and correlations.
Algorithm 3 Phase II with aR-tree indexing and 2 scans

Input: probabilistic relation \( R_p \), candidate set \( C_p \subseteq R_p \)

Output: Sky(\( R_p \))

1: build an aR-tree \( \text{indexTree} \) over \( R_p \) \( \triangleright \) Phase II.a
2: Sky(\( R_p \)) \( \leftarrow \) \( \emptyset \) \( \triangleright \) Phase II.b
3: for all tuples \( u \in C_p \) do
4: \hspace{1em} for all tuples \( v \in C_p \) preceding \( u \) do
5: \hspace{2em} if \( u \succ_p v \) [Rule 3] then \( C_p \leftarrow C_p \setminus \{v\} \)
6: \hspace{2em} else if \( v \succ_p u \) [Rule 3] then \( C_p \leftarrow C_p \setminus \{u\} \)
7: for all tuples \( u \in C_p \) do \( \triangleright \) 1st scan
8: \hspace{1em} insert \( \leftarrow \) true
9: \hspace{1em} for all tuples \( v \in \text{Sky}(R_p) \) do
10: \hspace{2em} if \( u \succ v \) then
11: \hspace{3em} \( \text{WndPr}(u,v) \leftarrow \text{indexTree}.\text{WndPr}(u,v) \)
12: \hspace{3em} if \( u \succ_p v \) [Rule 2] then Sky(\( R_p \)) \( \leftarrow \) Sky(\( R_p \)) \( \setminus \{v\} \)
13: \hspace{2em} else if \( v \succ u \) then
14: \hspace{3em} \( \text{WndPr}(v,u) \leftarrow \text{indexTree}.\text{WndPr}(v,u) \)
15: \hspace{3em} if \( v \succ_p u \) [Rule 2] then insert \( \leftarrow \) false, goto 16
16: if insert then \( \text{Sky}(R_p) \leftarrow \text{Sky}(R_p) \cup \{u\} \)

6.1 Experiments on the NBA Dataset

Our first series of experiments concerns the real NBA dataset: this dataset, that was also used in [20], contains 339,721 performances of 1,313 players during the seasons between 1991 and 2005. We only considered points, rebounds and assists, because other statistics (e.g., steals) are not available for each season. Each (unique) performance by a player is a tuple, while players represent groups: each player (group) has a total probability of 1, thus if he played in \( m \) games each of his performances has a probability of \( 1/m \) (if a same performance has been repeated \( k \) times, the probability of such tuple is \( k/m \)).

Our first experiment aims at giving an intuition about which tuples are included in the skyline of a probabilistic relation. To this end, we compare the skyline of the original dataset, with the skyline of datasets obtained by aggregating similar performances of each player into a single tuple. In the extreme case, where all performances by each player are “merged” into a single tuple, we obtain a dataset, where each player has a single tuple (with probability = 1) representing his “average” performance. It is therefore interesting to compare the skyline of datasets obtained between these two extreme cases (no aggregation vs. maximal aggregation). To this end, performances are first normalized into the [0,1] range and are then compared using the
Euclidean distance; player tuples whose distance is not higher than a cluster radius \( \varepsilon \) are combined into a single “average” tuple for that player.

Table 3 shows dataset and skyline sizes obtained for different \( \varepsilon \) values, when using both the expected rank and the expected score semantics. First of all, it is confirmed that the expected score semantics produces large skylines: on the original NBA dataset, the skyline size for expected score is about 23 times larger than the corresponding expected rank skyline. On the other hand, with the “average” dataset, we only have a single skyline for any ranking semantics, since such dataset involves no probability at all. It is therefore interesting to compare how the two different ranking semantics we consider here behave when varying the clustering radius \( \varepsilon \).

Table 3: NBA dataset and skyline size for the exp. rank and the exp. score semantics for different \( \varepsilon \) values

| \( \varepsilon \) | \( N \) | \( |\text{Sky}\left(\hat{R}^p\right)| \) |
|-------|------|------------------|
|       |      | exp. rank | exp. score |
| 0     | 231225 | 14       | 323        |
| 0.1   | 18924  | 91       | 668        |
| 0.2   | 5375   | 75       | 376        |
| 0.3   | 2703   | 39       | 256        |
| 0.4   | 1847   | 40       | 175        |
| 0.5   | 1516   | 29       | 110        |
| 0.6   | 1373   | 20       | 61         |
| 0.7   | 1331   | 19       | 37         |
| 0.8   | 1317   | 20       | 23         |
| 0.9   | 1313   | 20       | 20         |

As already noted before, the ranking of a tuple under the expected rank semantics is highly influenced by the value of its probability, i.e., tuples with a high probability have a higher rank than tuples with a low probability. In the NBA scenario, this means that, with \( \varepsilon = 0 \), the 14 tuples in the probabilistic skyline are all tuples with a high probability, i.e., good performances of players with a low number of played games. When increasing the clustering radius, however, such performances are dominated by average performances of outstanding players. This means that, for higher values of \( \varepsilon \), the probabilistic skyline under the expected rank semantics only contains performances of consistently good players, i.e., good performances with a high probability. For example, we analyze the case when \( \varepsilon = 0.5 \). This skyline contains performances of all the 20 best “average” players except Tim Hardaway, for which
two performances are now in the dataset: the cluster of his best performances has an 8% probability, while his most likely performance (92% probability) is not so good, being now dominated by the average performance of Stephon Marbury (which also has a higher probability). Clearly, performances of Tim Hardaway are not consistently outstanding, thus they are not included in the probabilistic skyline. On the other hand, this skyline contains performances from 10 other players, including Mike Bibby, Scottie Pippen, and Dwyane Wade. Although the average performance of such three players is not in the skyline, these are indeed outstandingly consistent players. For example, the average performance of Mike Bibby is dominated by that of Gary Payton. When, however, we reduce $\varepsilon$, performances of both players are divided into one average and high-probability cluster and one with low probability: the difference is that the low-probability cluster for Gary Payton contains very good performances, while Mike Bibby had a few bad games, thus the average (high-probability) performance of Gary Payton now no longer P-dominates that of Mike Bibby. The same happens for Scottie Pippen and Dwyane Wade which are now no longer dominated by the average performance of LeBron James.

When considering the expected score semantics, a tuple could be in the skyline because either its probability is high (as with the expected rank semantics) but also because its attribute values are high, even if its probability is low (see Equation ES:Rule 1). For instance, the skyline at $\varepsilon = 0.7$ includes outstanding performances, like the 81 points game by Kobe Bryant or the 34 rebounds game by Rony Seikaly. In both cases, such performances are the only game included in the cluster and represent the maximum value for an attribute (thus such performances are not dominated by any other performance). The expected score semantics, thus, introduces into the skyline outstanding (although possibly rare) performances, as opposed to the expected rank semantics which only includes consistent (highly probable) performances.

### 6.2 Experiments on Synthetic Datasets

The next series of experiments we present aims at analyzing the performance of algorithms presented in Section 5. In particular, we are interested in testing the scalability of algorithms and their robustness when varying the distribution of data. To this end, we performed experiments on efficiency on synthetic datasets, that allow to freely modify the different parameters used for data generation. In particular, we adapted to our scenario the procedure used in [25]. Once all the tuples have been generated according to a particular distribution, groups were formed as follows: a fraction $f_G$ of tuples is involved
in groups containing more than a tuple, while all other tuples will belong to “singleton” groups; each non-singleton group is formed by exactly \( N_G \) tuples, randomly picked from the dataset (this is repeated in case the probability of the group exceeds 1). We therefore have \( f_G N/N_G \) non-singleton groups and \((1 - f_G) N \) singletons. Table 4 shows the range of parameters used in the experiments; unless otherwise stated, default values are used.

Table 4: Settings for the generation of synthetic datasets

<table>
<thead>
<tr>
<th>symbol</th>
<th>description</th>
<th>range</th>
<th>default</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d )</td>
<td>dimensionality</td>
<td>([2 - 5])</td>
<td>4</td>
</tr>
<tr>
<td>( N )</td>
<td>cardinality</td>
<td>([10K - 200K])</td>
<td>100K</td>
</tr>
<tr>
<td>( N_G )</td>
<td>tuples per group</td>
<td>([2 - 6])</td>
<td>2</td>
</tr>
<tr>
<td>( f_G )</td>
<td>% of grouped tuples</td>
<td>([10% - 50%])</td>
<td>10%</td>
</tr>
</tbody>
</table>

For all experiments, we are mainly interested in overall clock time needed to compute the skyline. Our algorithms are implemented in Java\(^7\) and run on a Pentium IV 3.4 GHz PC equipped with 512 MB of main memory. Supplemental statistics include the number of candidate tuples and the effectiveness of P-domination rules. Due to the modular nature of our algorithms, each of them can be identified as a 4-part name using the symbols in Table 2. For instance, the algorithm \( \text{srt|R1|noidx|2scans} \) will perform a topological sort of tuples (\( \text{srt} \)), apply only Rule 1 in Phase I (\( \text{R1} \)), will not build any spatial index (\( \text{noidx} \)) and will perform 2 scans over candidates in Phase II (\( \text{2scans} \)).

In our first experiments we consider an uniform distribution for both data and probability.

We first compare alternatives for Phase I. The choices are whether to sort data and which rules to use to filter out tuples. Figure 5 shows actual running times for the 5 alternatives (\( \text{unrsr|R1R3}^+ \) is not a viable choice) when varying the data cardinality \( N \), the number of attributes \( d \), the number of tuples per group \( N_G \), and the \% of grouped tuples \( f_G \). Two main results are obtained from graphs in Figure 5: sorting tuples does not help in improving performances for \( \emptyset \) and \( \text{R1} \) alternatives (graphs are almost indistinguishable from those of the respective unsorted versions), whereas it is determinant in making \( \text{srt|R1R3}^+ \) consistently the best alternative for all the considered setups. When no rule is used, the quadratic dependence on \( N \) is evident in Figure 5 (a), whereas applying rules in Phase I largely mitigates this effect.

\(^7\)We thank Marios Hadjieleftheriou for his R-tree code.
On the other hand, performance of all strategies is largely independent of $N_G$ and $f_G$ (Figures 5 (c) and (d)): while this result was expected for unsrt|∅, since this strategy necessarily compares all pairs of tuples, it confirms that the other strategies are robust to modifications of these parameters.

![Graphs showing running times for alternative Phase I strategies](a) (b) (c) (d)

**Figure 5: Running times for alternative Phase I strategies**

It is also worth noting that, besides being the most efficient in computing bounds, the srt|R1R3+ variant is also able to filter out a high number of non-skyline tuples: Figure 6 shows the cardinality of Sky($R^p$) and $C^p$. In all experiments, the srt|R1R3+ strategy is able to eliminate 98% of P-dominated tuples during Phase I, reaching 99.9% for the simplest $d = 2$ dataset. This clearly also helps improving performance of Phase II, since only a limited number of candidate tuples are to be compared. Considering all the above, the next experiments will always adopt srt|R1R3+ for implementing Phase I.

The next experiment, shown in Figure 7, compares alternatives for Phase II.b, when no index is used. We only show results for varying $N$ and $d$.
because no visible effect is observed when changing other parameters. It is worth noting that both 1scan and 2scans strategies perform very well (running times are 1 order of magnitude lower than the time spent in Phase I) and that there is not much difference between them. 1scan is slightly better than 2scans for easier datasets (low $N$ and $d$ values), because in such cases the number of candidate tuples is very low and Rule 2 is checked only a few dozens times. On the other hand, for higher $N$ and $d$ values, the 2scans approach begins to pay off, saving about 20% of running time over 1scan when $d = 5$. In the following, therefore, we concentrate on the 2scans variant.

We finally analyze the alternatives for Phase II.a of the algorithm. Since we already opted for a sort in Phase I, the possible choices are whether an
index is useful and, in case, if an R-tree or an aR-tree is the best option. Figure 8 compares running times for the three alternatives. The noidx alternative is clearly always the best solution, outperforming both index-based solutions for all settings. However, it is interesting to note that, if we subtract from idx strategies the cost for building the tree, their running times are about 50% of the noidx approach: about 8 seconds are needed for computing the skyline in the default setting. Figures 8 (a) and (b) also show that R-tree is always slightly faster than aR-tree (besides being cheaper to build). This is due to the fact that an aR-tree is normally larger than an R-tree (due to the additional aggregated information stored in each tree node): it is likely that, in our scenario, this overhead is not balanced by the saving of accesses for nodes completely included within the search window. Finally, Figures 8 (c) and (d) again demonstrate that performance of considered strategies are unaffected when varying the $N_G$ and $f_G$ parameters.

![Figure 8](image-url)

Figure 8: Running times for alternative Phase II.a strategies
In our last series of experiments we examine performance of algorithms when varying the distribution of tuples and probability. In particular, we consider here data whose attributes are correlated positively or negatively, and also contemplate different distributions of probabilities. We first show, in Figure 9 (a), the cardinality of $\text{Sky}(R^p)$ for different data/probability distributions. The two letters in each dataset name correspond to data distribution (uniform, U, correlated, C, or anti-correlated, A) and probability distribution (uniform, U or Gaussian, G); numbers in brackets report the probability range for U or mean and variance for G.\footnote{We note here that, although, as proved by Theorem 3, the skyline of a probabilistic relation is independent of attribute values, it depends, at least for the expected rank semantics, on actual probability values.} The graphs also show the number of candidate tuples obtained when exploiting the $\text{srt}|R1R3^+|$ strategy during Phase I. It is clear that, for all considered distributions, the skyline size is of manageable size, reaching a maximum of 1832 tuples for the anti-correlated dataset; moreover, the limited number of candidate tuples again demonstrates the efficiency of the $\text{srt}|R1R3^+|$ strategy. Figure 9 (b) shows the effectiveness of the P-domination rules used by the $\text{srt}|R1R3^+|\text{noidx}|2\text{scans}$ algorithm for the different datasets: it is clear that the cheap Rule 1 is the most effective in filtering out P-dominated tuples. On the other hand, the costly Rule 2 is the least effective, amounting to around 0.02% of P-dominations in all cases, a fact that again proves the efficiency of our algorithm. Table 5, finally compares running times of the baseline $\text{unsr}|\emptyset|\text{noidx}|1\text{scan}$ algorithm with the most efficient $\text{srt}|R1R3^+|\text{noidx}|2\text{scans}$: for all data distributions, we obtain costs which are 1 order of magnitude lower than the naïve strategy, the most difficult dataset being, as usual, the anti-correlated one.

Table 5: Running times (in seconds) for the $\text{srt}|R1R3^+|\text{noidx}|2\text{scans}$ (and the $\text{unsr}|\emptyset|\text{noidx}|1\text{scan}$) algorithms

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Phase I</th>
<th>Phase II</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>UU[0.1]</td>
<td>156.2 (838.1)</td>
<td>17.1 (2106.0)</td>
<td>173.3 (2944.1)</td>
</tr>
<tr>
<td>CU[0.1]</td>
<td>130.4 (836.4)</td>
<td>0.3 (1470.7)</td>
<td>130.7 (2307.1)</td>
</tr>
<tr>
<td>AU[0.1]</td>
<td>202.3 (937.3)</td>
<td>137.1 (3522.3)</td>
<td>339.4 (4459.6)</td>
</tr>
<tr>
<td>UU[0.05]</td>
<td>227.8 (831.3)</td>
<td>20.9 (1801.9)</td>
<td>248.7 (2633.2)</td>
</tr>
<tr>
<td>UG[0.5,0.1]</td>
<td>176.6 (854.2)</td>
<td>27.0 (1895.5)</td>
<td>203.6 (2749.7)</td>
</tr>
<tr>
<td>UG[0.5,0.2]</td>
<td>181.7 (821.8)</td>
<td>16.4 (2524.8)</td>
<td>198.1 (3346.6)</td>
</tr>
<tr>
<td>UG[0.2,0.1]</td>
<td>124.4 (863.2)</td>
<td>20.9 (1562.7)</td>
<td>145.3 (2425.9)</td>
</tr>
</tbody>
</table>

We conclude our analysis on efficiency by showing results for the expected score semantics. As already stated in Section 5, in this case the
probabilistic skyline can be efficiently computed by appropriately modifying a “traditional” skyline algorithm; we show in Table 6 results obtained when applying the SFS algorithm [11] with sorting based on the sum of attributes and probability values.

Table 6: Running times (in milliseconds) and skyline size for the expected score semantics

| Dataset  | time (ms) | |Sky(Rp)| |
|----------|-----------|---------------------------------|
| UU[0,1]  | 104       | 860                             |
| CU[0,1]  | 35        | 75                              |
| AU[0,1]  | 641       | 2347                            |
| UU[0,0.5] | 100      | 860                             |
| UG[0.5,0.1] | 106   | 907                             |
| UG[0.5,0.2] | 129   | 1033                            |
| UG[0.2,0.1] | 108    | 889                             |

7 Conclusions

In this paper we have introduced an original definition of skyline for probabilistic relations that is able to preserve all the basic properties that skylines have in the deterministic scenario. In particular, the skyline is still equal to the union of all top-1 results when one considers all possible scoring functions that are monotone in the skyline attributes. We have also shown how domination among probabilistic tuples (P-domination), which lies at the heart of
our proposal, can be efficiently checked by means of a set of rules, which we have detailed for different ranking semantics. For efficiently computing the skyline we have provided a family of algorithms based on a 2-phase structure, and shown through experiments on large probabilistic datasets their improvement with respect to a naïve evaluation.

Our definitions are parametric in the semantics used for ranking probabilistic data, and we applied them to all the most notable semantics known in the literature. This nice property also allows for a new way of counterpointing ranking semantics, since one can now compare the skyline they induce over a same probabilistic relation. We plan to perform such analysis as soon as our implementation will also cover the case of the U-Topk, U-kRanks, and Global-Topk semantics.

References


