Chapter 2
Causality and Responsibility Problem on Probabilistic Reverse Skyline Queries

Abstract Causality and responsibility is an essential tool in the database community for providing intuitive explanations for answers/non-answers to queries. Causality denotes the causes for the answers/non-answers to queries, and responsibility represents the degree of a cause which reflects its influence on the answers/non-answers to queries. In this paper, we study the causality and responsibility problem (CRP) for the non-answers to probabilistic reverse skyline queries (PRSQ). We first formalize CRP on PRSQ, and then, we propose an efficient algorithm termed as CP to compute the causality and responsibility for the non-answers to PRSQ. CP first finds candidate causes, and then, it performs verification to obtain actual causes with their responsibilities, during which several strategies are used to boost efficiency. Extensive experiments demonstrate the effectiveness and efficiency of our presented algorithms.

Keywords Causality and responsibility · Probabilistic reverse skyline query Reverse skyline query · Algorithm

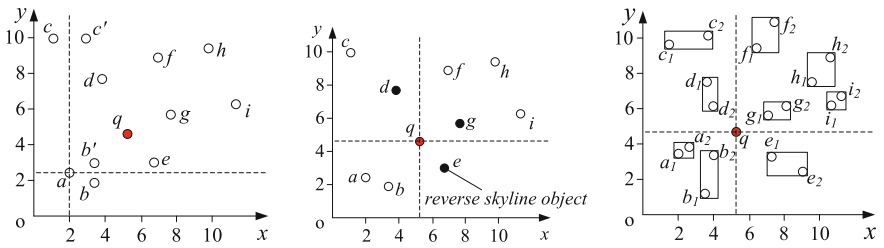
2.1 Introduction

Explanation capability is one of the important and essential features for database systems, which can offer users with the explanations for query results. Recently, causality and responsibility has become a useful tool in the database community for providing intuitive explanations for answers/non-answers to queries, which was first introduced to the database community by Meliou et al. [18–20, 22]. To be more specific, given a query Q over a database P, causality aims to find all the tuples in P that cause the presence of answers or the absence of non-answers to the query Q. Responsibility quantifies the effect that each cause has on the appearance of an answer or the absence of a non-answer, which is defined as a function of the size of the smallest contingency set. In the database literature, the *causality and responsibility problem* (CRP) has been explored in relational databases [18–20, 26] and probabilistic nearest neighbor search [17]. Nonetheless, CRP is *query-dependent*. None of the existing techniques can find efficiently the causality and responsibility

Y. Gao and Q. Liu, *Preference Query Analysis and Optimization*,
SpringerBriefs in Computer Science, https://doi.org/10.1007/978-981-10-6635-1_2

for the answers/non-answers to probabilistic reverse skyline queries (PRSQ), which has a wide range of applications such as multi-criteria decision making, sensor data monitoring, and business planning [1, 16]. To this end, in this chapter, we investigate CRP on probabilistic reverse skyline queries, which can enhance the explanation capability for database systems and thus improve the usability of the database. It is worth noting that in the chapter, we focus on the *non-answers*, because finding the causality and responsibility for the *answers* to probabilistic reverse skyline queries is relatively easy.

Before presenting the probabilistic reverse skyline query, we first give the definition of reverse skyline. For a specified D-dimensional dataset P and a query object q, the reverse skyline consists of all the objects whose dynamic skyline contains q [8]. In particular, if q belongs to the dynamic skyline of $p \in P$, there does not exist another object $p' \in P$ satisfying (i) $\forall \in [1, D]$, $|p'[i] - p[i]| \leq |q[i] - p[i]|$; and (ii) $\exists j \in [1, D]$, $|p'[j] - p[j]| < |q[j] - p[j]|$. Here, $p[i]$ refers to p's ith dimensional value, and we suppose the smaller the better. For example, in Fig. 2.1a, query object q does not belong to the dynamic skyline of a. Thus, a is not in the reverse skyline (i.e., d, e, and g) shown in Fig. 2.1b. Based on the reverse skyline, Lian and Chen [16] study the reverse skyline query on uncertain data, i.e., the probabilistic reverse skyline query, motivated by the uncertainty in real-world data. Given a D-dimensional uncertain dataset $\mathcal{P}$, a query object q, and a probability threshold α, a probabilistic reverse skyline query returns the objects in $\mathcal{P}$ whose probabilities to be reverse skyline objects are no less than α. Figure 2.1c depicts an example of probabilistic reverse skyline query, where each uncertain object has two samples with equal existence probability 0.5. The probability of an uncertain object u to be a reverse skyline object, denoted as $Pr(u)$, can be accumulated from all the possible worlds of $\mathcal{P}$ [16]. Hence, in Fig. 2.1c, $Pr(a) = 0.5$, $Pr(b) = 0.25$, $Pr(c) = 0$, $Pr(d) = 0.75$, $Pr(e) = 1$, $Pr(f) = 0.25$, $Pr(g) = 0.625$, $Pr(h) = 0$, and $Pr(i) = 0$. If $\alpha = 0.5$, uncertain objects a, d, e, and g form the probabilistic reverse skyline.



(a) Illustration of dynamic dominance (b) The reverse skyline query (c) The probabilistic reverse skyline query

Fig. 2.1 Example of reverse skyline and probabilistic reverse skyline queries

The probabilistic reverse skyline query is a useful tool for multi-criteria decision making [16]. As an example, the coach of a basketball team wants to recruit a new player with some preferred skills and selects candidates for the position. Clearly, a player is a candidate of this position if there are no other candidates who are more suitable to the position for all skills. Since the records of a basketball player are different in different seasons, the player can be modeled as an uncertain object, and all the records of the player constitute the instances of the uncertain object. Thus, in this case, the coach can take the new position as a query object and conduct a probabilistic reverse skyline query on the uncertain dataset formed by all basketball players to find those candidates who have the new position as their dynamic skylines with high probability.

In some instances, the returned results may disappoint the users. Continue the aforementioned example. If the basketball player finds himself absent from the candidate set, he might ask questions such as *What cause me unqualified for this position? What are the degrees of those causes?*, and is very keen to find out answers that might be helpful to him. Intuitively, if the player is not qualified for the position, there must have other more suitable players than him, which constitute the causes for his absence from the query result. Those causes enable the player to understand his competitors better and thus to improve his skills to exceed other players. To this end, we study the problem of finding the causality and responsibility for the non-answers to probabilistic reverse skyline queries. Take Fig. 2.1c as an example. It is observed that b is a non-probabilistic-reverse skyline object since the probability of q being dominated by a w.r.t. b is $0.75(> 0.5 = \alpha)$. Hence, a is the cause of the fact that b is a non-probabilistic-reverse skyline object. It is worth mentioning that the responsibility of a cause is defined as a function of the size of its smallest contingency set, which is to be detailed in Sect. 2.3.

Finding the causality and responsibility for the non-answers to probabilistic reverse skyline queries poses two major challenges. The first one is how to efficiently find the causes for the non-answers. To this end, we propose a filter-and-refinement framework to identify the causes for the non-answers to probabilistic reverse skyline queries. Specifically, we find the candidate cause set and then refine it to get the actual causes. The second challenge is how to efficiently find the minimum contingency set for every cause, since we define the responsibility of a cause as a function of the size of its smallest contingency set. Toward this, for each cause of a non-answer to the probabilistic reverse skyline query, the minimal contingency set is found by examining the candidate contingency set. In order to reduce the examination cost, we present several lemmas to identify the true objects (false objects) that must be present in (absent from) the minimum contingency set.

To sum up, the key contributions of this chapter are summarized as follows:

- We formalize the causality and responsibility problem (CRP for short) on probabilistic reverse skyline queries.
- We present an efficient algorithm to compute the CRP for the non-answers to probabilistic reverse skyline queries, based on the discrete sample and continuous pdf uncertain data models, respectively.

- We conduct extensive experiments to demonstrate the effectiveness and efficiency of our proposed algorithms.

The rest of this chapter is organized as follows. Section 2.2 reviews related work. Section 2.3 formulates the problem studied in the chapter. Sections 2.4 and 2.5 elaborate our approaches for computing the CRP for the non-answers to probabilistic reverse skyline queries on the discrete sample and continuous pdf uncertain data models, respectively. Section 2.6 reports experimental results and our findings. Finally, Sect. 2.7 concludes the chapter.

2.2 Related Work

Causality is originally an active research area in the field of logic and philosophy over the centuries [6, 11, 15, 23]. Recently, Meliou et al. [18–20] extended the notions of causality and responsibility to the database community. They present a general overview of causality in the context of databases [18], introduce functional causality as a refined definition of causality [18], and compute the causes and their responsibilities for conjunctive queries [19]. Meliou et al. [21] propose the view-conditioned causality to account for the effect of a tuple on multiple outputs (views). Lian and Chen [17] investigate the causality and responsibility for probabilistic nearest neighbor search in uncertain databases in order to handle the sensitivity of query answers. Qin et al. [26] explore the computation of the responsibility for the lineages of conjunctive queries with inequalities. Moreover, Freire et al. [9] study how particular interventions, e.g., tuple deletions in the input of a query, impact its output. It is worth mentioning that causality and responsibility are query-dependent. The current efforts mostly focus on SQL queries and probabilistic nearest neighbor search. Thus, existing causality and responsibility techniques cannot be applied directly to tackle the causality and responsibility computation on probabilistic reverse skyline queries. It is necessary to design efficient approaches to find the causality and responsibility for the non-answers to probabilistic reverse skyline queries.

Data provenance, which explores the derivation of a piece of data that is in the query result, has been widely studied in the database literature. The data provenance models include why-provenance, where-provenance, how-provenance. To compute data provenance efficiently, many methods have been proposed, including non-annotation approaches [3, 7] and annotation methods [2, 5]. The non-annotation approaches create an inverted query to compute the data provenance, whereas the annotation methods utilize extra information, which is recorded during the evaluation of a query, to derive the provenance of data. In addition to certain databases, data provenance in uncertain databases has also been investigated in [12, 13, 28, 29].

As pointed out by Meliou et al. [19], data provenance is related to why-so causality, i.e., the causality for answers. Nonetheless, in this chapter, we focus on the causality for non-answers. Therefore, it is infeasible to utilize data provenance techniques to our work. In addition, we also take the responsibility of causes into consideration, which is ignored by data provenance.

Table 2.1 Symbols and descriptions

Notation	Description
$\mathcal{P}$	an uncertain D-dimensional dataset
P	a certain D-dimensional dataset
$\rho(o, a_n)$	the responsibility of a cause o for a non-answer a_n
Γ	a contingency set
$P \vDash (\nvDash) RSQ(a)$	the object a is (not) an answer to the reverse skyline query over a dataset P
$\mathcal{P} \vDash (\nvDash) PRSQ(a)$	the object a is (not) an answer to the probabilistic reverse skyline query on an uncertain dataset $\mathcal{P}$
$Pr(u)$	the probability of an uncertain object u to be a reverse skyline point
α	a probability threshold
$pw(\mathcal{P})$	a possible world of an uncertain dataset $\mathcal{P}$

2.3 Problem Statement

In this section, we formalize the causality and responsibility problem (CRP) over probabilistic reverse skyline queries. Table 2.1 summarizes the notations used frequently throughout this chapter.

2.3.1 CRP Formulation

Let P be a D-dimensional dataset and Q be a query. The fact that an object $a \in P$ is (is not) an answer to Q over P is denoted as $P \vDash Q(a)$ ($P \nvDash Q(a)$). Next, we formally define the causality and responsibility for non-answers to queries below.

Definition 2.1 (*Causality*) Given a D-dimensional dataset P, a non-answer a_n to a query Q, i.e., $P \nvDash Q(a_n)$, and an object $p(\neq a_n) \in P$. Then, for the query Q over P: (i) If $(P - p) \vDash Q(a_n)$, p is a counterfactual cause for a_n; (ii) If there exists a contingency set $\Gamma \subseteq P$ for p such that $(P - \Gamma) \nvDash Q(a_n)$ but $(P - \Gamma - p) \vDash Q(a_n)$, p is an actual cause for a_n.

In other words, if a non-answer a_n becomes an answer after removing an object p from P, the object p is a counterfactual cause for a_n. Furthermore, if there is a contingency set $\Gamma \subseteq P$ satisfying that p is a counterfactual cause for a_n on $P - \Gamma$, the object p is an actual cause for a_n. Note that the contingency set actually consists of the tuples from the dataset P, i.e., it is the subset of P, and there might be many contingency sets for a real cause. It is obvious that the counterfactual cause is a special case of an actual cause where $\Gamma = \varnothing$.

It is worth mentioning that the definition of causality follows [17], which is different from the one defined for relational databases [19]. This is because the causality defined in [19] is to find the objects, denoted as O, whose absence from the data-

base P causes a_n to be a non-answer, i.e., if we add O to P, a_n will become the answer. However, for the probabilistic reverse skyline query, whether a_n object is a non-answer to the probabilistic reverse skyline query is only determined by the database $\mathcal{P}$, rather than the objects that are not in $\mathcal{P}$. Consequently, the definition of the causality proposed by Meliou et al. [19] cannot be applied directly to our studied problem. Based on the definition of causality, we define the responsibility below, which follows [17, 19].

Definition 2.2 (*Responsibility*) Let an object $p \in P$ be an actual cause for a non-answer a_n to a query Q, and Γ range over all contingency sets for p. Then, the responsibility of p for a_n, denoted as $\rho(p, a_n)$, is:

$$\rho(p, a_n) = \frac{1}{1 + min_{\Gamma}|\Gamma|} \tag{2.1}$$

In brief, the responsibility of p being a cause for a non-answer is defined as a function of the size of p's smallest contingency set. According to Definition 2.2, if an object is a counterfactual cause for a non-answer, its responsibility is 1 due to $\Gamma = \varnothing$, and if an object is an actual cause for a non-answer, its responsibility is between 0 and 1. By convention, if an object is not a cause for a non-answer, its responsibility is 0. Based on Definitions 2.1 and 2.2, we summarize the causality and responsibility problem as follows:

Causality and Responsibility Problem (CRP) Given a D-dimensional dataset P, and a non-answer a_n to a query Q on P, the causality and responsibility problem (CRP) on Q should compute the set of the actual causes from P, denoted as C, together with their corresponding responsibilities for a_n.

2.3.2 CR2PRSQ Formulation

In this subsection, we first formally define the reverse skyline query (RSQ) [8] and the probabilistic reverse skyline query (PRSQ) [16], and then, we present the definition of the CRP on PRSQ. Given three objects p_1, p_2, and p_3 in a D-dimensional dataset P, if p_1 *dominates* p_2 w.r.t. p_3, denoted as $p_1 \prec_{p_3} p_2$, it must hold that (i) $\forall i \in [1, D]$, $|p_1[i] - p_3[i]| \leq |p_2[i] - p_3[i]|$, and (ii) $\exists j \in [1, D]$, $|p_1[j] - p_3[j]| < |p_2[j] - p_3[j]|$ [24].

Definition 2.3 (*Reverse Skyline Query* [8]) Given a D-dimensional dataset P and a query object q, a reverse skyline query (RSQ) finds all the objects in P that take q as one of their dynamic skyline objects, that is, if an object $p \in P$ is a reverse skyline object of q, there does not exist any other object $p'(\neq p) \in P$ such that $p' \prec_p q$.

For instance, Fig. 2.1b shows an example of RSQ, where objects d, e, and g constitute the reverse skyline of a specified query object q.

Based on the reverse skyline query, we formalize the probabilistic reverse skyline query. First, we introduce the uncertain data model for uncertain datasets. Specifically, given a D-dimensional uncertain dataset $\mathcal{P}$, every uncertain object $u \in \mathcal{P}$ is modeled by an uncertain region, denoted as $UR(u)$, in which u resides. The probabilistic distribution of u is described by either *discrete samples* [14, 25] or *a continuous probability density function* (pdf) [4, 27]. For ease of understanding, in the sequel, we assume that the uncertain dataset follows the discrete sample model. Nevertheless, all the techniques proposed for the CRP on PRSQ can also be extended to the pdf model, as to be discussed later. For the discrete sample model, each uncertain object $u \in \mathcal{P}$ contains l_u mutually exclusive samples $u_i (1 \leq i \leq l_u)$. Every sample u_i is assigned with an appearance probability $u_i.p$ satisfying (i) $0 \leq u_i.p \leq 1$ and (ii) $\sum_{i=1}^{l_u} u_i.p = 1$. Like work presented in [16, 17], we assume that uncertain objects in the dataset are independent of each other, and the coordinates of every object are also independent. Now, we formally define the probabilistic reverse skyline query.

Definition 2.4 (*Probabilistic Reverse Skyline Query* [16]) Given a D-dimensional uncertain dataset $\mathcal{P}$, a query object q, and a probability threshold $\alpha \in (0, 1]$, a probabilistic reverse skyline query (PRSQ) retrieves those objects $u \in \mathcal{P}$ such that the probability of u being a reverse skyline object, denoted as $Pr(u)$, is no smaller than α, i.e.,

$$Pr(u) = \sum_{i=1}^{l_u} u_i.p \cdot \left(\prod_{\forall u' \in \mathcal{P} - \{u\}} \left(1 - Pr\{u' \prec_{u_i} q\}\right)\right) \geq \alpha \tag{2.2}$$

in which $Pr\{u' \prec_{u_i} q\}$ is the probability of q being dynamically dominated by u' w.r.t. u_i, and

$$Pr\{u' \prec_{u_i} q\} = \sum_{j=1 \wedge u'_j \prec_{u_i} q}^{l_{u'}} u'_j.p \tag{2.3}$$

For simplicity, we assume that the query object q is a certain object, following [16, 17]. According to Definition 2.4, we formulate the CRP on PRSQ below. Note that we call a non-answer to a probabilistic reverse skyline query as a non-probabilistic-reverse skyline object.

Definition 2.5 (*CRP on PRSQ*) Given an uncertain dataset $\mathcal{P}$, a query object q, a probability threshold $\alpha \in (0, 1]$, and a non-probabilistic-reverse skyline object a_n, the CRP on PRSQ (CR2PRSQ) needs to (i) find a set $C \subseteq P$ such that (a) $\forall c \in C$, c is an actual cause for a_n, and (b)$\forall c' \in (P - C)$, c' is not an actual cause for a_n; and (ii) $\forall c \in C$, compute its degree of responsibility $\rho(c, a_n)$ based on Eq. (2.1).

As an example, in Fig. 2.1c , the uncertain object c is a non-probabilistic-reverse skyline object as $Pr(c) = 0 (\alpha = 0.5)$. If the uncertain object d is deleted from the uncertain dataset, $Pr(c) = 1$, i.e., c becomes a probabilistic reverse skyline object. Therefore, the uncertain object d is a counterfactual cause for c, with responsibility $\rho(d, c) = 1$.

We would like to highlight that CR2PRSQ and CR-PNN problem studied in [17] are very different. The goal of CR2PRSQ is to compute all the causes and their responsibilities for a specified non-probabilistic-reverse skyline object while the target of CR-PNN problem is to compute the *expected responsibility*, which is different from responsibility, for each uncertain object on probabilistic nearest neighbor queries.

2.4 CP Algorithm

In this section, we present the algorithm for computing the causality and responsibility for a non-probabilistic-reverse skyline object. In what follows, we employ a running example, shown in Fig. 2.2, to facilitate the understanding of CR2PRSQ computation. Specifically, it contains an uncertain object set $P = \{a, b, c, d, e, f, g, h, i\}$, in which each uncertain object has two through four samples. For simplicity, we assume that all the samples corresponding to the same uncertain object share equal existence probabilities. In other words, if an uncertain object u has l_u samples, each sample $u_i \in u(i \in [1, l_u])$ has its existence probability being $1/l_u$. In addition, we assume that the uncertain object c is a specified non-probabilistic-reverse skyline object, and the probability threshold $\alpha = 0.5$.

CR2PRSQ computation involves two aspects, i.e., the computation of the causality and its corresponding responsibility. Based on Definition 2.2, the responsibility is defined as a function of the size of the smallest contingency set, which can be found during the computation of causality. Thus, in the following, we will mostly focus on how to find the causality for a given non-probabilistic-reverse skyline object.

A naive method to support CR2PRSQ computation is to examine, for every uncertain object $u \in \mathcal{P}$, all the subsets of the uncertain dataset $\mathcal{P}$ to find u's contingency set. Clearly, this approach has $O(|\mathcal{P}| \times 2^{|\mathcal{P}|})$ as its time complexity, in which $|\mathcal{P}|$ refers to the cardinality of $\mathcal{P}$. It is obvious that this naive approach is infeasible because of

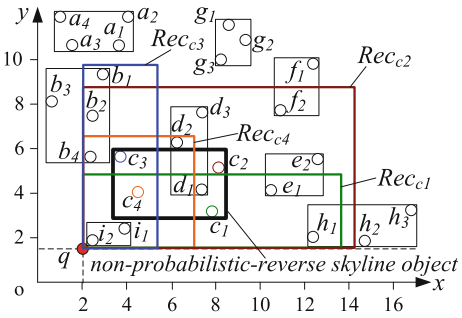


Fig. 2.2 A running example of CR2PRSQ (© [2016] IEEE. Reprinted, with permission, from [10])

high time complexity. Consequently, next, we propose a more efficient algorithm for computing the causality and responsibility for the specified non-probabilistic-reverse skyline object.

CR2PRSQ aims to find all the causes for the non-probabilistic-reverse skyline object. As mentioned in the naive method for CR2PRSQ earlier, the whole search space is $\mathcal{P}$, which is very large especially when the cardinality of $\mathcal{P}$ is high. Fortunately, we find that some objects cannot be the actual causes for the non-probabilistic-reverse skyline object, and identifying and removing such objects can help to shrink the search space. In the sequel, we first present Lemma 2.1 to identify those unqualified causes.

Lemma 2.1 *Given an uncertain dataset $\mathcal{P}$, a non-probabilistic-reverse skyline object a_n, an uncertain object $u \in \mathcal{P}$, and a query object q, if q is not dominated by u w.r.t. a_n in all possible worlds of $\mathcal{P}$, u is not an actual cause for a_n.*

Proof Since a_n is a non-probabilistic-reverse skyline object, based on Definition 2.4, there must exist other object(s) in $\mathcal{P}$, denoted as $\mathcal{O} \subseteq \mathcal{P}$, such that $\forall o \in \mathcal{O}$, the probability of q being dynamically dominated by o w.r.t. a_n is bigger than 0. Note that the set $\mathcal{O}$, rather than $\mathcal{P} - \mathcal{O}$, determines the $Pr(a_n)$, i.e., the probability of an being a probabilistic reverse skyline object. If the uncertain object u is an actual cause for a_n, it must satisfy one of the conditions listed in Definition 2.1. (a) If we remove u from the dataset $\mathcal{P}$, a_n is still a non-answer to the probabilistic reverse skyline query on the dataset $\mathcal{P} - \{u\}$ (i.e., $(\mathcal{P} - \{u\}) \nvDash PRSQ(a_n)$), as $\mathcal{O}$ does not change and the $Pr(a_n)$ also does not change. Hence, it does not satisfy the condition (i) of Definition 2.1, meaning that u is not a counterfactual cause for a_n. (b) Assume that there is a non-empty set $\Gamma (\subseteq \mathcal{P})$. If $\Gamma \supset \mathcal{O}$, $(\mathcal{P} - \Gamma) \vDash PRSQ(a_n)$ (i.e., a_n is an answer to the probabilistic reverse skyline query on the dataset $\mathcal{P} - \Gamma$); otherwise, $(\mathcal{P} - \Gamma - \{u\}) \nvDash PRSQ(a_n)$. Consequently, there is no qualifying contingency set for u that satisfies the condition (ii) of Definition 2.1, and it is not a real cause for a_n. Therefore, u is not an actual cause for a_n, and the proof completes. □

In other words, if q is dominated by u w.r.t. a_n with the probability 0, the uncertain object u cannot be an actual cause for a_n. Based on this observation, we can find all the candidate causes for a non-probabilistic-reverse skyline object with the time complexity of $O(|\mathcal{P}|^2)$. That is to compute the probability of q being dominated by u w.r.t. a_n, for every uncertain object $u \in \mathcal{P}$. However, we are not satisfied with $O(|\mathcal{P}|^2)$ time complexity, and we would like to introduce a more efficient method that does not need to traverse the entire dataset $\mathcal{P}$.

Lemma 2.2 *Given two uncertain objects $u, u' \in P$, and a query object q, for each sample $u_i \in u$, we form a hyper-rectangle Rec_i that is centered at u_i and has the coordinate-wise distance to the query object q as its extent. If there is a sample $u'_k \in u' (i \neq k)$ locating within the hyper-rectangle Rec_i formed by a sample $u_i \in u$, the probability of q being dominated by u' w.r.t. u is bigger than 0.*

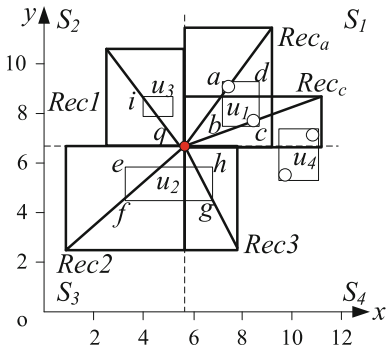


Fig. 2.3 Example of Lemma 2.2 (© [2016] IEEE. Reprinted, with permission, from [10])

Proof If a sample $u'_k \in u'$ locates inside the hyper-rectangle formed by the sample $u_i \in u$, it must hold that (i) $\forall j \in [1, D]$, $|u'_k[j] - u_i[j]| \leq |q[j] - u_i[j]|$; and (ii) $\exists j \in [1, D]$, $|u'_k[j] - u_i[j]| < |q[j] - u_i[j]|$. Thus, there is a possible world, in which q is dominated by u' w.r.t. u. According to Eq. (2.3), the probability of q being dominated by u' w.r.t. u is the summation of all the possible worlds, where q is dominated by u' w.r.t. u. Hence, the probability of q being dominated by u' w.r.t. u is larger than 0, and the proof completes. □

Take Fig. 2.3, which is in a 2-dimentional space, as an example. One of the samples of uncertain object u_4 lies within the rectangle Rec_c formed by a sample of u_1. Thus, q has the probability to be dominated by u_4 w.r.t. u_1. Based on Lemma 2.2, we can find efficiently all the candidate causes by using the range query. To be more specific, we first maintain a hyper-rectangle list to store all the hyper-rectangles formed by every sample of a non-probabilistic-reverse skyline object. Then, we traverse the R-tree, which indexes the uncertain dataset $\mathcal{P}$, in a branch-and-bound manner. If the sample of an uncertain object locates inside one of the hyper-rectangles in the hyper-rectangle list, the corresponding uncertain object is one of the candidate causes for the non-probabilistic-reverse skyline object. Otherwise, it is not a candidate cause. Take the running example in Fig. 2.2 as an example. In Fig. 2.2, we form the rectangle for each sample of c, namely Rec_{c_1}, Rec_{c_2}, Rec_{c_3}, and Rec_{c_4}. Obviously, uncertain objects b, d, e, f, h, and i all have the sample(s) falling into at least one of the rectangles formed by c_1, c_2, c_3, and c_4. Thus, b, d, e, f, h, and i constitute the candidate causes for c.

After finding the candidate cause set, denoted as C_c, we need to further examine every $c_{c_i} \in C_c$ by finding a contingency set $\Gamma \in \mathcal{P}$, such that (i) $(\mathcal{P} - \Gamma) \not\models PRSQ(a_n)$ [i.e., a_n is not an answer to the probabilistic reverse skyline query over an uncertain dataset $(\mathcal{P}-\Gamma)$]; and (ii) $(\mathcal{P}-\Gamma-\{c_{c_i}\}) \models PRSQ(a_n)$ [i.e., a_n is an answer to the probabilistic reverse skyline query on an uncertain dataset $\mathcal{P} - \Gamma - \{c_{c_i}\}$]. If we can find the Γ satisfying the above two conditions, the candidate cause c_{c_i} is an actual cause for a_n. Note that if c_{c_i} is an actual cause for a_n, we need to find the minimum Γ among all the contingency sets in order to compute the responsibility of c_{c_i}. In other words, if we manage to find the minimum contingency set earlier, we

can achieve both the goal of determining whether the candidate cause is an actual cause and the goal of computing the responsibility of the cause earlier.

In the following, we explain how to find the minimum contingency set for the candidate cause. As $\Gamma \subseteq \mathcal{P}$, a naive approach is to examine all the subsets of $\mathcal{P}$. However, in total $2^{|\mathcal{P}|}$ subsets have to be examined for a single candidate cause in the worst case. Evidently, it is too expensive, and we prefer to examine a smaller number of subsets. To this end, we develop Lemma 2.3 below, which guarantees that some objects in $\mathcal{P}$ cannot contribute to the minimum contingency set and thus can be skipped.

Lemma 2.3 *Given an uncertain dataset $\mathcal{P}$, a non-probabilistic-reverse skyline object a_n, a candidate cause set C_c, an actual cause $c \in C_c$ for a_n with the corresponding minimum contingency set Γ, and an uncertain object u, if $u \in \mathcal{P} - C_c$, $u \notin \Gamma$.*

Proof Assume, to the contrary, that $u \in \Gamma$. Since c is an actual cause for a_n with the minimum contingency set Γ, (i) $(\mathcal{P} - \Gamma) \not\models PRSQ(a_n)$, and (ii) $(\mathcal{P} - \Gamma - \{c\}) \models PRSQ(a_n)$. According to Eq. (2.2), the probability $Pr(a_n)$ of an being a reverse skyline object is only impacted by the objects in C_c. As the object $u \in \{\mathcal{P} - C_c\}$, u does not affect $Pr(a_n)$. Hence, (i) $(\mathcal{P} - \Gamma - \{u\}) \not\models PRSQ(a_n)$, and (ii) $(\mathcal{P} - \Gamma - \{c\} - \{u\}) \models PRSQ(a_n)$, indicating that $\Gamma - \{u\}$ is also a contingency set for the actual cause c. However, $|\Gamma - \{u\}| < |\Gamma|$. Thus, Γ is not the minimum contingency set, which contradicts the condition of Lemma 2.3. The proof completes. □

Based on Lemma 2.3, only the objects in the candidate cause set C_c can contribute to the minimum contingency set, i.e., $\Gamma \subseteq C_c$. For instance, in Fig. 2.2, uncertain objects a and g are not the candidate causes. Thus, they can be discarded safely when finding the minimum contingency set for any actual cause. Using Lemma 2.3, we can prune away unqualified objects to reduce the search space for the minimum contingency set. In addition, we also observe that some objects in C_c are definitely present in (absent from) the minimum contingency set for any actual cause, as stated in the following lemmas.

Lemma 2.4 *Given an uncertain dataset $\mathcal{P}$, a non-probabilistic-reverse skyline object a_n, a candidate cause set C_c, an actual cause $c \in C_c$ for a_n with the corresponding minimum contingency set Γ, and an uncertain object $c'(\neq c) \in C_c$, for every sample of a_n, we can form a hyper-rectangle centered at p and having the coordinate-wise distance to a query object q as its extent, if c' is contained in all the hyper-rectangles formed by each sample of a_n, then $c' \in \Gamma$.*

Proof Assume, to the contrary, $c' \notin \Gamma$. Based on Eq. (2.2), if $\mathcal{P}$ contains c', $Pr(a_n) = 0$. Hence, (i) $(\mathcal{P} - \Gamma) \not\models PRSQ(a_n)$ and (ii) $(\mathcal{P} - \Gamma - \{c\}) \not\models PRSQ(a_n)$, meaning that Γ is not a qualified contingency set, which contradicts the condition of Lemma 2.4. Therefore, $c' \in \Gamma$, and the proof completes. □

Lemma 2.5 *Given an uncertain dataset $\mathcal{P}$, a non-probabilistic-reverse skyline object a_n, an actual cause c_1 for an with the corresponding minimum contingency*

set Γ, and an uncertain object $c_2(\neq c_1) \in \mathcal{P}$, if c_2 is a counterfactual cause for a_n, then $c_2 \notin \Gamma$.

Proof Assume, to the contrary, $c_2 \in \Gamma$. Since c_2 is a counterfactual cause for a_n, $(\mathcal{P} - c_2) \models PRSQ(a_n)$. If $c_2 \in \Gamma$, $(\mathcal{P} - \Gamma) \models PRSQ(a_n)$. Hence, Γ is not a qualified contingency set of c_1, which contradicts with the condition of Lemma 2.5. Thus, $c_2 \notin \Gamma$, and the proof completes. □

Based on Lemmas 2.4 and 2.5, we have that (i) if an uncertain object u is contained in all the hyper-rectangles formed by the samples of a_n, u must be present in the minimum contingency set of any other actual cause and (ii) if an uncertain object u' is a counterfactual cause, it must be absent from the minimum contingency set of any other actual cause. The two lemmas enable us to further narrow the search space of the contingency set and the number of the subsets to be examined can be reduced to $2^{|C_c - C_a \cup C_b|}$, where C_a denotes the set of the objects that locate into the hyper-rectangles formed by all the samples of a specified non-probabilistic-reverse skyline object, and C_b represents a counterfactual cause set. Nonetheless, if $|C_c - C_a \cup C_b|$ is still large, the processing time is still costly. Consequently, we develop Lemma 2.6 to further cut down the number of the subsets to be examined.

Lemma 2.6 *Given an uncertain dataset $\mathcal{P}$, an actual cause c for an with the corresponding minimum contingency set $\Gamma(\neq \varnothing)$, and a candidate cause $c' \in \Gamma$, if $(\mathcal{P} - \{\Gamma - \{c'\}\} - \{c\}) \not\models PRSQ(a_n)$, c' is an actual cause for a_n with the contingency set $(\{\Gamma - \{c'\}\}) \cup \{c\}$.*

Proof As c is a real cause for a_n, $(\mathcal{P} - \Gamma - \{c\}) \models PRSQ(a_n)$. If $(\mathcal{P} - \{\Gamma - \{c'\}\} - \{c\}) \not\models PRSQ(a_n)$, it satisfies the condition (ii) of Definition 2.1. Thus, c' is an actual cause for a_n with the contingency sets $(\{\Gamma - \{c'\}\}) \cup \{c\}$. □

Actually, the minimum contingency set Γ of an actual cause c can also be used to shrink the search space of the minimum contingency set for other candidate causes in Γ according to Lemma 2.6. Moreover, recall that our goal is to find the minimum contingency set for the candidate cause. If we examine the sets in the order of their cardinalities, the first contingency set Γ found will be the minimum one. These observations explain how we further reduce the number of the candidate contingency sets examined, and how we terminate the evaluation process earlier.

Integrating all the aforementioned techniques, we develop an algorithm, termed as CP, to compute the causality and responsibility for the non-probabilistic-reverse skyline object, which follows a filter-and-refinement framework. Specifically, CP utilizes Lemma 2.2 to get the candidate causes for a given non-probabilistic-reverse skyline object, and then, it gets the actual causes and their responsibilities using Lemmas 2.3, 2.4, 2.5, and 2.6. Algorithm 1 presents the pseudo-code of algorithm CP.

CP takes as inputs a query object q, a non-probabilistic-reverse skyline object a_n, an uncertain object set $\mathcal{P}$ indexed by an R-tree, and a probability threshold α, and outputs all the actual causes and their responsibilities for a_n. In the filtering step, CP

finds all the candidate causes by traversing the R-tree $\mathcal{R}_{\mathcal{P}}$ (lines 1–9). Specifically, CP forms a hyper-rectangle list $RecList$ for a_n (line 1). Then, it accesses the R-tree in a branch-and-bound manner. For an entry e that intersects with a hyper-rectangle in $RecList$, if e is a data object, it is added to the candidate cause set; otherwise, e must be an intermediate node and thus is expanded. After finding all the candidate causes, if $\alpha = 1$, all the candidate causes are the actual causes with equal responsibility (lines 10–12). Otherwise, the algorithm (i) finds the objects that must be in any causes contingency set and adds them to Γ_1 (line 16) and (ii) finds all the counterfactual causes and removes them from C_c since they cannot be in the minimum contingency set of other actual causes (lines 17–18). Next, CP initializes $n_i \in N$ with $|C_c| - 1$, which records the cardinality of the minimum contingency set currently found for the remaining candidate causes (lines 19–20). In the refinement step, CP invokes the function **FMCS** to find the minimum contingency set for every remaining candidate cause with the cardinality smaller than n_i (line 22). If the minimum contingency set exists, the candidate cause is a real cause, and FMCS computes its responsibility (lines 23–24). Note that if $n_i \neq |C_c| - 1$ and there does not exist a contingency set whose cardinality is smaller than n_i, the candidate cause is also an actual cause (lines 25–26), because the minimum contingency set is found by FMCS based on Lemma 2.6 before the candidate cause is verified.

The pseudo-code of FMCS is shown in Algorithm 2. It retrieves the minimum contingency set via examining sets based on ascending order of their cardinalities (lines 2–4). It is terminated as soon as function **combine** returns the minimum contingency set Γ (lines 9–18). Then, FMCS uses Γ to shrink the search space of the contingency set for other candidate causes in Γ (lines 5–7). Finally, FMCS returns the minimum contingency set. It is worth noting that **combine** is a recursive function to return the subset of the candidate cause set (lines 9–15), and to verify whether it is a contingency set (lines 16–17).

Continue the running example in Fig. 2.2 . Objects b, d, e, f, h, and i are returned as the candidate causes in the filtering step. Since i falls into all the hyper-rectangles formed by c, it must be present in the contingency set for other actual causes. In the refinement step, we need to examine every candidate cause. Take object b as an example. CP examines the subsets of $\{d, e, f, h\}$ in ascending order of their cardinalities. After examining, b's minimum contingency set is found, i.e., $\Gamma_b = \{d, i\}$. Thus, $\rho(b, c) = 1/3$. Then, the algorithm uses the object b and Γ_b to examine d and i. We can find the contingency sets for d and i, i.e., $\Gamma_d = \{b, i\}$ and $\Gamma_i = \{d, b\}$, which are also the minimum contingency sets for d and i. Hence, $\rho(d, c) = \rho(i, c) = 1/3$, and the examinations for d and i are saved. Next, CP proceeds to examine the following candidate causes until all of them are examined. Finally, we can get all actual causes for c, i.e., $\{\{b, \rho(b, c) = 1/3\}, \{d, \rho(d, c) = 1/3\}, \{e, \rho(e, c) = 1/3\}, \{f, \rho(f, c) = 1/4\}, \{h, \rho(h, c) = 1/3\}, \{i, \rho(i, c) = 1/3\}\}$.

Algorithm 1 CR2PRSQ Algorithm (CP)

Input: a non-probabilistic-reverse skyline object a_n, a query object q, an R-tree $\mathcal{R}_\mathcal{P}$ on a set $\mathcal{P}$ of uncertain data objects, a threshold α
Output: a causality and responsibility set CR

form the hyper-rectangles for each sample of a_n, and store them in $RecList$
initialize the min-heap H with all the root entries of $R_\mathcal{P}$
while H is not empty **do**
  de-heap the top entry e of H
  if e crosses a $rec \in RecList$ **then**
    if e is a data object **then**
      add e to C_c
    else
      expand e, and insert all its child entries to H
if $\alpha = 1$ **then**
  for each entry $c_{c_i} \in C_c$ **do**
    add c_{c_i} to CR, and $c_{c_i}.r \leftarrow 1/|C_c|$
else
  for each entry $c_{c_i} \in C_c$ **do**
    if c_{c_i} locates into each hyper-rectangle in $RecList$ **then**
      add c_{c_i} to Γ_1
    if c_{c_i} is a counterfactual cause for a_n **then**
      add c_{c_i} to CR, $c_{c_i}.r \leftarrow 1$, and delete c_{c_i} from C_c
  for each $n_i \in N$ **do**
    $n_i \leftarrow |C_c|$¨$C1$
  for each entry $c_{c_i} \in C_c$ **do**
    $\Gamma \leftarrow$ **FMCS**$(C_c, c_{c_i}, n_i, \Gamma, \Gamma_1, \alpha)$
    if $\Gamma \neq \varnothing$ **then**
      add c_{c_i} to CR, and $c_{c_i}.r \leftarrow 1/(|\Gamma| + 1)$
    if $\Gamma = \varnothing$ and $n_i \neq |C_c| - 1$ **then**
      add c_{c_i} to CR, and $c_{c_i}.r \leftarrow 1/(n_i + 1)$
return CR

2.5 Discussion

Up to now, we have proposed algorithm CP to compute the causality and responsibility for the non-probabilistic-reverse skyline object on the discrete sample model. In the sequel, we extend CP to the continuous pdf model. Obviously, CP can be easily extended to tackle the CRP on probabilistic reverse skyline queries under the continuous pdf model. Nonetheless, there are three differences we would like to mention.

First, in the filtering step, algorithm CP finds the candidate causes for a non-probabilistic-reverse skyline object using a range query. Under the pdf model, it is impossible to form the hyper-rectangles for all the samples of a non-probabilistic-reverse skyline object. Instead, for an uncertain object u following the pdf model, we only need to form a single hyper-rectangle via the farthest object to q in the uncertain region of u. For instance, in Fig. 2.3, i is the farthest object in the uncertain region of u_3 to q. Hence, $Rec1$ is the hyper-rectangle formed by u_3. However, if an

Algorithm 2 Finding Minimal Contingency Set (FMCS)

Input: a candidate cause set C_c, a candidate cause c_c to be verified, the cardinality of currently found minimal contingency sets n_i of c_c, a set Γ to store the minimal contingency set, a set Γ_1 that must be contained in the final Γ, a threshold α
Output: the minimal contingency set Γ

1: initialize $\Gamma \leftarrow \varnothing$
2: **for** $i \leftarrow 1$ to $n_i - |\Gamma_1|$ **do**
3: $tag \leftarrow$ **combine**$(C_c - \Gamma_1 - \{c_c\}, \Gamma_1, i, \Gamma)$
4: **if** *tag* is True **then break**
5: **for** each entry $o_j \in \Gamma$ **do**
6: **if** o_j is not verified and $(\mathcal{P} - \{\Gamma - \{o_j\}\} - \{c_c\}) \nvDash PRSQ(a_n)$ **then**
7: $n_j \leftarrow |\Gamma|$
8: **return** Γ
Function combine(C, Γ_1, i, Γ)
9: $tag \leftarrow False$
10: **for** $j \leftarrow 1$ to $|C|Ci$ **do**
11: add $c_j \in C$ to Γ
12: **if** $i > 1$ **then**
13: **combine**$(C - \{c_j\}, \Gamma_1, i - 1, \Gamma)$
14: **else**
15: $\Gamma \leftarrow \Gamma \cup \Gamma_1$
16: **if** $(\mathcal{P} - \Gamma) \nvDash PRSQ(a_n)$ and $(\mathcal{P} - \Gamma - \{c_c\}) \vDash PRSQ(a_n)$ **then**
17: $tag \leftarrow true$
18: **return** tag

uncertain object u lies in different sub-quadrates formed by q, the hyper-rectangle of u is the union of the hyper-rectangles formed in every sub-quadrate. Take Fig. 2.3 as an example again. Uncertain object u_2 has its uncertain region located into two sub-quadrates, i.e., sub-quadrate 3 (S_3) and sub-quadrate 4 (S_4). Based on the farthest object to the query object q, $Rec2$ and $Rec3$ are the rectangles of u_2 formed in S_3 and S_4, respectively. Thus, the final rectangle of u_2 is the union of $Rec2$ and $Rec3$ (i.e., $Rec2 \cup Rec3$).

Second, algorithm CP needs to find the objects that are located in all the actual causes' minimum contingency sets (Algorithm 1, line 15). For the pdf model, if the objects fall into the hyper-rectangle formed by the nearest objects to q in the uncertain region of the non-probabilistic-reverse skyline object, these objects must be in every actual cause's minimum contingency set. For example, in Fig. 2.4, the shadow is the hyper-rectangle formed by the closest objects to q in the uncertain region of u_1. If an uncertain object locates inside the shaded area, it must be contained in the minimal contingency set of u_1's actual cause. Note that if an object falls into many sub-quadrants, it cannot form such hyper-rectangle, e.g., the object u_2 in Fig. 2.4.

Third, algorithm CP needs to compute the probability of an uncertain object being a reverse skyline object to judge whether it is a probabilistic reverse skyline object (Algorithm 2, line 6 and line 16). The probability is the summation of all possible

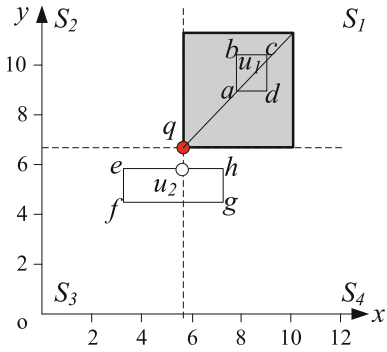


Fig. 2.4 Example of the pdf model (© [2016] IEEE. Reprinted, with permission, from [10])

worlds under discrete sample model, while that under the pdf model is the integration of the whole uncertain object.

In the following, we analyze the time complexity of algorithm CP. Let $|\mathcal{RP}|$ be the cardinality of the R-tree indexing an uncertain dataset $\mathcal{P}$, $|C_c|$ be the cardinality of a candidate cause set, $|C_a|$ be the cardinality of the objects that must be in the minimum contingency of real causes, and $|C_b|$ be the cardinality of a counterfactual cause set. The time complexity of algorithm CP is presented in Theorem 2.1 below.

Theorem 2.1 *The time complexity of algorithm CP is* $O(|\mathcal{R}_\mathcal{P}| + |C_c| \times 2^{|C_c - C_a \cup C_b|})$.

Proof Algorithm CP can be divided into two steps, i.e., filtering and refinement. In the filtering step, CP traverses the R-tree to get candidate causes. In worst case, it takes $O(|\mathcal{R}_\mathcal{P}|)$. In the refinement step, algorithm CP verifies the candidate causes by examining the subsets of the candidate cause set. For each candidate cause, it needs to examine $2^{|C_c - C_a \cup C_b|}$ subsets in the worst case. Thus, it needs $O(|C_c| \times 2^{|C_c - C_a \cup C_b|})$ time in the refinement step. Totally, the time complexity of algorithm CP is $O(|\mathcal{R}_\mathcal{P}| + |C_c| \times 2^{|C_c - C_a \cup C_b|})$. The proof completes. □

2.6 Performance Study

In this section, we experimentally evaluate the effectiveness and efficiency of our proposed algorithms under a variety of experimental settings. All experiments are conducted in a Windows PC with 2.8 GHz CPU and 4 GB main memory, and all the algorithms are coded in C++.

2.6.1 Experimental Setup

In our experiments, we use synthetic datasets, to verify the performance of algorithm CP. We generate synthetic uncertain datasets with various parameter values, similar

Table 2.2 Parameter ranges and default values

Parameter	Range
Dimensionality d	2, **3**, 4, 5
Dataset cardinality	10K, 50K, **100K**, 500K, 1000K
Probability threshold α	0.2, 0.4, **0.6**, 0.8, 1
The range of radius $[r_{min}, r_{max}]$	[0, 2], [0, 3], **[0, 5]**, [0, 8], [0, 10]

as [16, 17]. In order to produce an uncertain object u, we first select the center C_u of u in a D-dimensional data space within domain [0, 10000] for each dimension. Then, we select a radius r within $[r_{min}, r_{max}]$, which indicates the maximum deviation of the object position from C_u. In addition, we create randomly a hyper-rectangle that is tightly bounded by the sphere centered at the center C_u with radius r. We consider two classes of distributions for C_u, i.e., *Uniform* and *Skew*. We utilize two types of radius distributions for r, i.e., *Uniform* and *Gaussian*. Within the uncertain region of each object, we generate random samples following the uniform distribution. Therefore, we have four types of datasets, denoted as *lUrU*, *lUrG*, *lSrU*, and *lSrG*, where *lU* and *lS* denote the distribution of center C_u following *Uniform* and *Skew*, respectively; *rU* and *rG* represent the distribution of radius r following *Uniform* and *Gaussian*, respectively.

We study the performance of our presented algorithms under various parameters. Their value ranges and default settings are listed in Table 2.2, which follows [8, 16, 17]. We deploy the number of node accesses (i.e., I/O) and CPU time as the main performance metrics. In the experiments, we select randomly 50 non-answers and report their average performance.

2.6.2 Experimental Results

In this section, we demonstrate the effectiveness and efficiency of algorithm CP. First, we compare the performance of CP with naive algorithm. As mentioned earlier, the time complexity of naive method for CR2PRSQ is $O(|\mathcal{P}| \times 2^{|\mathcal{P}|})$, which is extremely expensive. Hence, we present an improved baseline for CR2PRSQ here, denoted as Naive-I. Naive-I first finds the candidate causes like CP, and then, it refines them by examining all the subsets of candidate cause set. Figure 2.5 depicts the performance of the two algorithms. It is observed that the I/O of CP and Naive-I is the same, while the CPU time of algorithm CP outperforms that of Naive-I algorithm. This is because the I/O cost mainly comes from the first step of the algorithms, i.e., finding the candidate causes, which is the same for both CP and Naive-I. Consequently, the I/O cost of them is identical. At the refinement step, algorithm CP utilizes a series of strategies to boost efficiency. Thus, the CPU time of algorithm CP is smaller than that of Naive-I algorithm.

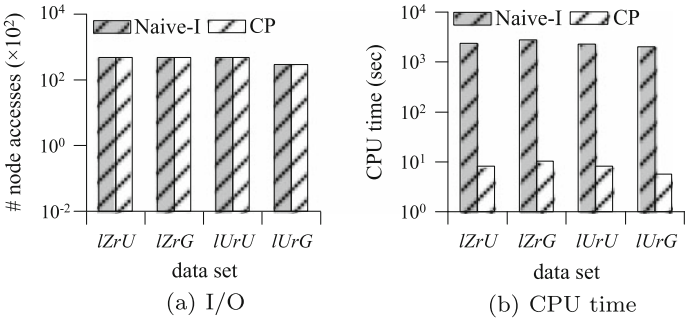


Fig. 2.5 CP cost versus naive cost

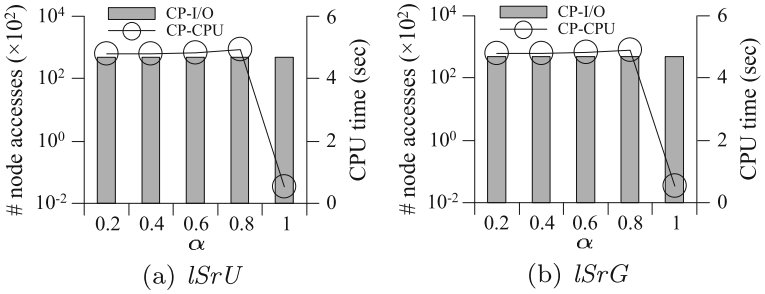


Fig. 2.6 CP cost versus α

Second, we explore the impact of probability threshold α on algorithm CP. We use synthetic datasets with $|\mathcal{P}|$ = 100K, dimensionality = 3, and the range of radius $[r_{min}, r_{max}]$ being [0, 5], and report the performance of CP in Fig. 2.6. It is observed that the CPU time of algorithm CP degrades with the growth of α while the number of node accesses remains the same. CP follows the filter-and-refinement framework. In the filtering step, it traverses the R-tree to find the candidate causes for a non-probabilistic-reverse skyline object. All the node accesses are made during the filtering step. Although the probability threshold varies, the non-probabilistic-reverse skyline object does not change. Therefore, the number of node accesses of CP remains unchanged. Notice that the CPU time of the filtering step does not change. In the refinement step, CP finds the minimum contingency set Γ for every candidate cause. When α becomes larger, the cardinality of Γ increases as well. Hence, it takes more time in the refinement step as α ascends. Accordingly, the CPU time of CP also grows. However, when $\alpha = 1$, the CPU time of algorithm CP drops dramatically. This is because, when $\alpha = 1$, all the candidate causes are the final result, and thus, the refinement step is skipped which helps to cut down CPU time.

Third, we study the influence of the range of radius $[r_{min}, r_{max}]$ on algorithm CP, and present the results in Fig. 2.7. It is observed that the performance of CP degrades as $[r_{min}, r_{max}]$ increases. The reason is that if the range of radius becomes larger,

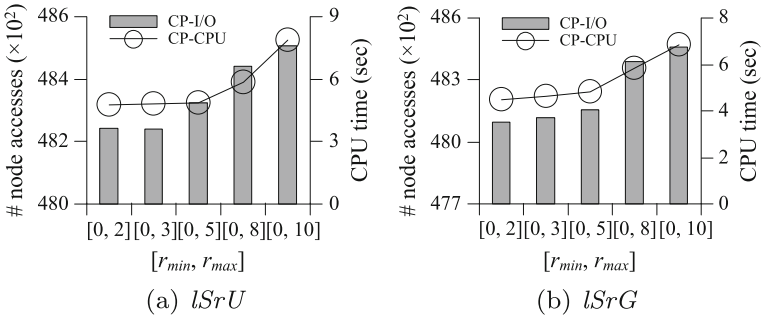


Fig. 2.7 CP cost versus $[r_{min}, r_{max}]$

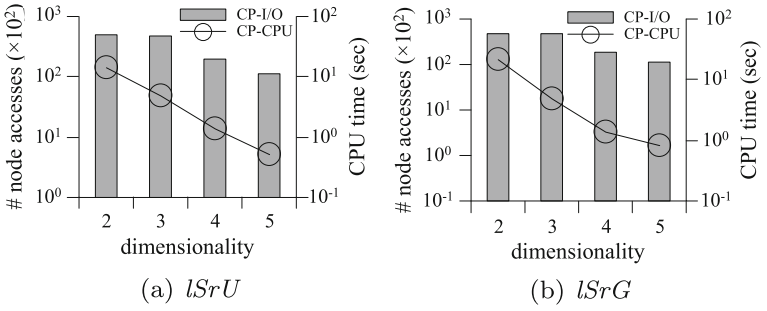


Fig. 2.8 CP cost versus dimensionality

the hyper-rectangle formed by the samples of the non-probabilistic-reverse skyline object is enlarged accordingly. Hence, the number of candidate causes ascends, and algorithm CP takes more time in both filtering and refinement steps.

Next, we investigate the impact of dimensionality. Toward this, we vary dimensionality from 2 to 5, and fix $\alpha = 0.6$, $[r_{min}, r_{max}] = [0, 5]$, and $|\mathcal{P}| = 100$K. As expected, in Fig. 2.8, the number of node accesses and CPU time drop as dimensionality increases. This is because, in the high dimension, uncertain objects are dominated by fewer objects. Therefore, the number of the actual causes for the non-probabilistic-reverse skyline object is decreased, and the performance of algorithm CP is improved.

Finally, we inspect the impact of cardinality on algorithm CP. Figure 2.9 plots the corresponding results. As expected, the I/O cost and CPU time of CP ascend as $\mathcal{P}$ grows. The reason is that the larger the dataset cardinality is, the more intensive the data is. Thus, there are more candidate causes for the non-probabilistic-reverse skyline object, incurring longer processing time.

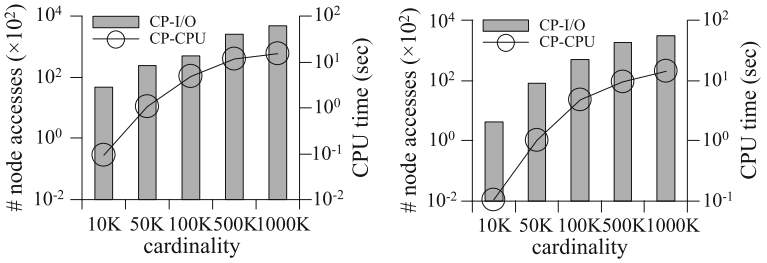


Fig. 2.9 CP cost versus cardinality

2.7 Conclusion

In this chapter, for the first time, we study the problem of the causality and responsibility on probabilistic reverse skyline queries. We propose an algorithm called CP to efficiently compute the causality and responsibility for the non-probabilistic-reverse skyline object. Specifically, CP follows the filter-and-refinement framework and utilizes several strategies to speed up the computation. Extensive experiments demonstrate the performance of our presented algorithms. In the future, we plan to develop more efficient algorithms for CR2PRSQ computation. Also, we intend to investigate the CRP on other queries, such as reverse top-k queries.

References

1. Bai, M., Xin, J., Wang, G.: Probabilistic reverse skyline query processing over uncertain data stream. In: DASFAA, pp. 17–32 (2012)
2. Bhagwat, D., Chiticariu, L., Tan, W.C., Vijayvargiya, G.: An annotation management system for relational databases. VLDB Journal **14**(4), 373–396 (2005)
3. Buneman, P., Khanna, S., Tan, W.C.: Why and where: A characterization of data provenance. In: ICDT, pp. 316–330 (2001)
4. Cheng, R., Kalashnikov, D.V., Prabhakar, S.: Evaluating probabilistic queries over imprecise data. In: SIGMOD, pp. 551–562 (2003)
5. Chiticariu, L., Tan, W.C., Vijayvargiya, G.: Dbnotes: A post-it system for relational databases based on provenance. In: SIGMOD, pp. 942–944 (2005)
6. Chockler, H., Halpern, J.Y.: Responsibility and blame: A structural-model approach. Journal of Artificial Intelligence Research **22**, 93–115 (2004)
7. Cui, Y., Widom, J.: Lineage tracing for general data warehouse transformations. VLDB Journal **12**(4), 41–58 (2003)
8. Dellis, E., Seeger, B.: Efficient computation of reverse skyline queries. In: VLDB, pp. 291–302 (2007)
9. Freire, C., Gatterbauer, W., Immerman, N., A.Meliou: The complexity of resilience and responsibility for self-join-free conjunctive queries. PVLDB **9**(3), 180–191 (2015)
10. Gao, Y., Liu, Q., Chen, G., Zhou, L., Zheng, B.: Finding causality and responsibility for probabilistic reverse skyline query non-answers. IEEE Trans. Knowl. Data Eng **28**(11), 2974–2987 (2016)

11. Halpern, J.Y., Pearl, J.: Causes and explanations: A structural-model approach. part i: Causes. British Journal for the Philosophy of Science **56**(4), 843–887 (2005)
12. Kanagal, B., Deshpande, A.: Lineage processing over correlated probabilistic databases. In: SIGMOD, pp. 675–686 (2010)
13. Kanagal, B., Li, J., Deshpande, A.: Sensitivity analysis and explanations for robust query evaluation in probabilistic databases. In: SIGMOD, pp. 841–852 (2011)
14. Kriegel, H.P., Kunath, P., Renz, M.: Probabilistic nearest-neighbor query on uncertain objects. In: DASFAA, pp. 337–348 (2007)
15. Lewis, D.: Causation. The Journal of Philosophy **70**(17), 556C567 (1973)
16. Lian, X., Chen, L.: Acm trans. database syst. ACM Trans. Database Syst. **35**(1), 3 (2010)
17. Lian, X., Chen, L.: Causality and responsibility: Probabilistic queries revisited in uncertain databases. In: CIKM, pp. 349–358 (2013)
18. Meliou, A., Gatterbauer, W., Halpern, J.Y., C. Koch, K.F.M., Suciu, D.: Causality in databases. IEEE Data Eng. Bull. **33**(3), 59–67 (2010)
19. Meliou, A., Gatterbauer, W., Moore, K.F., Suciu, D.: The complexity of causality and responsibility for query answers and non-answers. PVLDB **4**(1), 34–45 (2010)
20. Meliou, A., Gatterbauer, W., Moore, K.F., Suciu, D.: Why so? or why no? functional causality for explaining query answers. In: MUD, pp. 3–17 (2010)
21. Meliou, A., Gatterbauer, W., Nath, S., Suciu, D.: Tracing data errors with view-conditioned causality. In: SIGMOD, pp. 505–516 (2011)
22. Meliou, A., Roy, S., Suciu, D.: Causality and explanations in databases. PVLDB **7**(13), 1715–1716 (2014)
23. Menzies, P.: Counterfactual theories of causation. Stanford Encylopedia of Philosophy (2008)
24. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive skyline computation in database systems. ACM Trans. Database Syst. **30**(1), 41C82 (2005)
25. Pei, J., Jiang, B., Lin, X., Yuan, Y.: Probabilistic skylines on uncertain data. In: VLDB, pp. 15–26 (2007)
26. Qin, B., Wang, S., Zhou, X., Du, X.: Responsibility analysis for lineages of conjunctive queries with inequalities. IEEE Trans. Knowl. Data Eng. **26**(6), 1532–1543 (2014)
27. R. Cheng, D.V.K., Prabhakar, S.: Querying imprecise data in moving object environments. IEEE Trans. Knowl. Data Eng **16**(9), 1112–1127 (2004)
28. Re, C., Suciu, D.: Approximate lineage for probabilistic databases. PVLDB **1**(1), 797–808 (2008)
29. Sarma, A.D., Theobald, M., Widom, J.: Exploiting lineage for confidence computation in uncertain and probabilistic databases. In: ICDE, pp. 1023–1032 (2008)

http://www.springer.com/978-981-10-6634-4

Preference Query Analysis and Optimization

Gao, Y.; Liu, Q.

2017, XI, 110 p. 58 illus., 23 illus. in color., Softcover
ISBN: 978-981-10-6634-4